REGLAMENTO INTERNO

PROGRAMA DOCTORADO EN INGENIERÍA INFORMÁTICA

Aprobado por el Consejo de Departamento de Informática

el 3 de septiembre de 2021

Dada la naturaleza del trabajo académico y en pos de un mejoramiento continuo, el presente reglamento será revisado y sancionado por el CPCT anualmente. Si se registraren cambios esenciales, éstos aplicarán solamente a nuevas cohortes de estudiantes.

INTRODUCCIÓN

Art. 1 El programa de **Doctorado en Ingeniería Informática** fue creado el 11 de julio del 2002, por acuerdo del Consejo Superior de la UTFSM en su sesión Nº 127, como consta en la resolución Acuerdo Nº 442.

Art. 2 El programa de Doctorado en Ingeniería Informática, en adelante **DII**, se regirá por estas normas y por el Reglamento General Nº 47 de Estudios de Postgrado en la Universidad Técnica Federico Santa María (UTFSM).

Art. 3 Estas normas complementan el Reglamento General Nº 47, en todas aquellas materias no contempladas en él, o que se han establecido allí, expresamente, como materias a ser reguladas por normas específicas de cada programa de postgrado.

TÍTULO I: DISPOSICIONES GENERALES

Art. 4 **Objetivos del Programa**: El objetivo del DII es formar especialistas que deseen adquirir conocimientos avanzados en el área de la Ingeniería Informática, a través de un programa sistemático de cursos y seminarios, y la realización de una investigación teórica o aplicada conducente a la tesis doctoral, de modo que, a través de dicha investigación original e independiente, contribuya al desarrollo científico de la Ingeniería Informática.

Art. 5 **Áreas de especialización del Programa**: El DII ofrece especialización en:

a) Computación Aplicada.
b) Inteligencia Artificial y Ciencia de los Datos.
c) Software y Ciberseguridad.

Art. 6 **Perfil del graduado**: El graduado del DII debe ser capaz de:

a) Desarrollar una investigación pura o aplicada, en el área de la ingeniería informática.
b) Analizar un problema de investigación específico en alguna área de la ingeniería informática, estudiar el estado del arte asociado a este problema, y proponer una solución original e independiente.
c) Evaluar la calidad o el desempeño de la solución propuesta al problema de investigación.
d) Resumir y difundir los resultados obtenidos durante la investigación y divulgarlos en revistas y conferencias internacionales de su especialidad.
e) Proponer y ejecutar proyectos de investigación en su área de la disciplina.

Art. 7 El doctorante deberá cumplir una permanencia activa mínima en el DII equivalente a 150 SCT en la UTFSM (2,5 años) en régimen de jornada completa.

Art. 8 La permanencia máxima en el DII, para un alumno regular, es de 6 años.

TITULO II: ADMINISTRACIÓN DEL PROGRAMA

Art. 9 La tuición académica del programa DII dependerá, de manera exclusiva, del Departamento de Informática (en adelante **DI**) de la UTFSM.

Art. 10 El DII será dirigido por el Comité Académico del DII (en adelante **CADII**), integrado por al menos cuatro miembros del Cuerpo de Directores de Tesis del Programa (en adelante **CDTP**) definido en el Título III, pertenecientes a las dos jerarquías académicas o docentes superiores, y será presidido por el Director del Programa, definido en el Art.12. Todos ellos serán designados por el Consejo del DI a propuesta del Director del DI (Anexo Nº 1).

Art. 11 Le corresponde al CADII, además de las funciones establecidas en el Art. 16 del Reglamento General Nº 47:

a) Programar cursos y designar sus profesores.
b) Aprobar, en primera instancia, programas de nuevas asignaturas o cualquier otra modificación del plan de estudios.
c) Actualizar, periódicamente, el Cuerpo de Profesores (Art.15) y el Cuerpo de Directores de Tesis del Programa (Art.16), de acuerdo a los criterios establecidos en el Título III.
d) Sesionar al menos 3 veces al año, quedando sus acuerdos consignados en actas según el reglamento de sala del DII, sin perjuicio de las atribuciones del Consejo del DI.
e) Aplicar los mecanismos de evaluación del DII establecidos.
f) Participar en las actualizaciones de los planes de desarrollo del DI.
g) Exponer ante el cuerpo de profesores del DII las situaciones de conflicto académico o disciplinario que se presentaren, para una adecuada resolución.
h) Otras competencias o actos, de índole académico, necesarios para la buena marcha del programa DII, corresponden al Director del Programa.

Art. 12 El Director del DII será un miembro del CADII, designado por el Consejo del DI a propuesta del Director del DI.

Art. 13 Las sesiones del CADII serán convocadas por el Director del DII. Este Comité deberá sesionar con la concurrencia de más de la mitad de sus miembros. En caso de que alguna materia deba decidirse por votación, se aplicará el criterio de mayoría absoluta de los miembros. De producirse un empate en la votación, decidirá el Director del DII.

Art. 14 El Director del DII designará entre los demás miembros del CADII un Director Alterno para subrogarlo en su ausencia.

TITULO III: DE LOS PROFESORES DEL PROGRAMA

Art. 15 Podrán pertenecer al Cuerpo de Profesores (CP-DII) del DII, académicos y docentes de la UTFSM que tengan el grado de Doctor y que estén en una de las tres jerarquías académicas o docentes superiores. Excepcionalmente, y por acuerdo fundado del CADII, podrán pertenecer al CP-DII investigadores o profesores de la Universidad u otras Instituciones que tengan reconocida trayectoria en un área disciplinar del DII. Quienes deseen integrar el CP-DII deberán manifestarlo explícitamente.

Art. 16 Conformarán el Cuerpo de Directores de Tesis del Programa (CDTP) los integrantes del CP-DII que cumplieren con las siguientes exigencias de productividad científica:

a) Índice de **productividad promedio anual** de artículos en revistas indexadas por WoS igual o superior a **1,4** para los últimos **5 años**.

b) Se considerará para el cálculo del índice de productividad, las publicaciones en Revistas indexadas por WoS (con **ponderación 1,0**) y Proceedings indexados en WoS o Scopus (con **ponderación 0,5**), con al menos **6** de estas publicaciones en Revistas indexadas por WoS.

Art. 17 Los integrantes del CP-DII que no cumplan con lo estipulado en el Art. 16 serán Colaboradores del programa.

Art. 18 Excepcionalmente, y por acuerdo fundado del CADII, podrá ser Co-Director de Tesis Doctoral un profesor o investigador que no pertenezca al CDTP, pero que tenga reconocida trayectoria en una línea de investigación relacionada con el tema de tesis específico. En tal caso, el CADII designará un Director de Tesis entre los miembros del CDTP.

Art. 19 Podrán participar en el DII, y por acuerdo del CADII, profesores o investigadores de otras Universidades o Centros de Investigación que tenga reconocida trayectoria en un área disciplinar del DII. Estos profesores o investigadores participarán en calidad de Profesor Visitante del programa.

Art. 20 Los miembros del CDTP que dejen de cumplir las exigencias de productividad científica establecidas en el Art.16, no podrán guiar tesis adicionales, pero continuarán dirigiendo aquellas Tesis Doctorales que ya hubieren sido aprobadas en los correspondientes Exámenes de Calificación.

TITULO IV: DE LA ADMISIÓN

Art. 21 Las postulaciones se canalizarán a través de la Dirección de Postgrado y Programas (en adelante DPP), y deberán seguir los procedimientos y formalidades establecidos en el Reglamento General Nº 47. Será requisito esencial para postular al DII tener el grado de licenciado en ciencias de la ingeniería o en ciencias. El CADII podrá autorizar excepcionalmente, mediante resolución fundada, el ingreso de personas que se encuentren en trámites finales de graduación o situaciones académicas que lo ameriten, estando su aceptación definitiva supeditada a la obtención del grado correspondiente, de conformidad al procedimiento que al efecto apruebe el CADII.

Art. 22 Una solicitud de admisión será analizada por el CADII sólo cuando el postulante haya hecho llegar todos los antecedentes requeridos, establecidos en el procedimiento de admisión de la DPP.

Art. 23 Una vez enviados los antecedentes completos del postulante por la DP, el Director del DII los pondrá a disposición del CADII, el que resolverá por mayoría absoluta sobre la aceptación o rechazo de la postulación. Para tomar la decisión, el CADII considerará las calificaciones de asignaturas de pregrado, carta de interés del postulante, cartas de recomendación, publicaciones en revistas y conferencias, participación en proyectos científico-tecnológicos, y otras consideraciones académicas.

Art. 24 Un postulante será admitido al DII sólo si el CDTP incluye un especialista en la línea de investigación de su interés, cuidando que exista un adecuado equilibrio entre el número de estudiantes aceptados y el total de recursos disponibles.

Art. 25 El CADII, o el Director del Programa, podrá exigir que el postulante entregue antecedentes adicionales y/o que participe en una entrevista, presencial o remota, para evaluar aspectos relevantes que permitan decidir en mejor forma sobre la solicitud de admisión.

Art. 26 El CADII podrá exigir que el postulante apruebe una etapa de magister, antes de ser aceptado en el DII.

Art. 27 Una vez que el CADII resuelva sobre la aceptación o rechazo del postulante al DII, el Director del DII informará dicha decisión a la DPP. Si el postulante es admitido en el DII, el CADII designará un Tutor entre los miembros del CDTP, cuyo nombre también será informado a la DPP.

Art. 28 El Tutor propondrá al CADII un Plan de Estudios específico para el doctorante, el que podrá incluir la convalidación u homologación de hasta 30

SCT en asignaturas del Programa de Estudios. El CADII deberá aprobar el Plan de Estudios y convalidaciones, o pedir cambios. Cuando fuere aceptado, será informado a la DPP.

TITULO V: SOBRE EL DESARROLLO DEL PROGRAMA

Art. 29 Cada doctorante deberá aprobar al menos 60 créditos SCT del DII en Asignaturas de nivel postgrado (Programa de Estudios), y un Trabajo de Tesis (Actividad de Graduación) equivalente a 180 créditos SCT, correspondientes a los Seminarios de Tesis (Anexo Nº 3).

Art. 30 Las asignaturas del DII serán evaluadas con nota de 0 a 100, siendo 70 la nota mínima de aprobación (Anexo Nº 4).

Art. 31 Una vez que el estudiante haya aprobado los créditos de asignaturas, deberá informar al Director del DII el nombre del miembro del CDTP que será su Director de Tesis. Esta información deberá ser confirmada por escrito por el Director de Tesis propuesto, quien a partir de ese momento será también el Tutor del estudiante en el DII. El Director del DII comunicará esta decisión a la DPP y al Tutor previo, si los hubiere.

Art. 32 Durante la realización de la tesis, el alumno de doctorado deberá inscribir las asignaturas Seminario de Tesis I, de 60 créditos SCT, y Seminario de Tesis II, de 120 créditos SCT. El profesor de estas asignaturas será el Director de Tesis del alumno.

TITULO VI: EL EXAMEN DE CALIFICACIÓN

Art. 33 El Examen de Calificación estará basado en la defensa de su proyecto de tesis doctoral y en evaluación de conocimientos relevantes en los temas fundamentales de su especialidad, atingentes al proyecto de tesis doctoral, adquiridos por el doctorante durante el desarrollo del Programa de Estudios. El doctorante que apruebe este examen será Candidato a Doctor.

Art. 34 El tema de tesis deberá ser propuesto por el estudiante y su Director de Tesis en el formato establecido por el CADII, el que estará disponible en la página web del Programa. Dicha propuesta deberá llevar la firma del estudiante y del Director de Tesis, y será entregada al Director del DII, al menos 15 días hábiles antes del Examen de Calificación.

Art. 35 El Examen de Calificación será rendido ante la Comisión de Examen de Calificación, nominada específicamente por el CADII, y que estará integrada por:

a) El Director de Tesis.
b) Un Examinador Interno, miembro del CDTP designado por el CADII.
c) Uno o más Examinadores Externos, experto en el área externo a la UTFSM, designado por el CADII.
d) El Presidente de la Comisión, con derecho a voz, que será el Director del DII o quien éste designe entre los miembros del CADII, no pudiendo recaer esta responsabilidad en el Director de Tesis.

Art. 36 El Director del DII deberá hacer llegar el proyecto de Tesis Doctoral a los miembros de la Comisión de Examen de Calificación, con una antelación de, al menos, 10 días hábiles.

Art. 37 En caso de reprobación, este Examen podrá repetirse sólo una vez, dentro de un plazo de 6 meses, previo acuerdo de la Comisión. En caso de reprobar por segunda vez, el estudiante será eliminado del DII.

Art. 38 El Examen de Calificación deberá ser aprobado por cada doctorante durante sus dos primeros años de permanencia en el DII.

Art. 39 El Seminario de Tesis I será calificado en escala de 0-100, una vez que el doctorante haya aprobado su Examen de Calificación.

Art. 40 El Seminario de Tesis II será calificado en escala de 0-100, una vez que el Candidato a Doctor haya culminado su trabajo de tesis doctoral y se encuentre en condiciones de rendir su Examen de Grado.

TITULO VII: DE LA TESIS DE DOCTORADO Y EXAMEN DE GRADO

Art. 41 El Comité de Tesis evaluará el Trabajo de Tesis y el Examen de Grado, y estará integrado por:

a) El Director de Tesis.
b) Un Examinador Interno, miembro del CDTP designado por el CADII.
c) Al menos dos Examinadores Externos, que sean Profesores o Investigadores externos a la UTFSM, expertos en el área, designados por el Comité de Coordinación y Desarrollo de Postgrado de la UTFSM, a proposición del CADII.
d) El Presidente de la Comisión, con derecho a voz, que será el Director del DII o quién éste designe entre los miembros del CADII, no pudiendo recaer esta responsabilidad en el Director de Tesis.

Art. 42 El Examen de Grado será público, y consistirá en una presentación y defensa oral de la Tesis por parte del doctorante.

Art. 43 El Trabajo de Tesis consistirá en un trabajo personal de investigación en la línea de especialidad del estudiante; el que deberá conformar un cuerpo

escrito novedoso y significativo de conocimientos y generar, al menos, una publicación en alguna revista indexada por WoS.

La publicación deberá estar aceptada antes de la entrega del trabajo de Tesis.

Además, el Candidato deberá adjuntar una declaración simple de originalidad, independencia, exclusividad y reproducibilidad de los antecedentes contenidos en su Trabajo de Tesis (Formato Declaración Simple en Anexo Nº 6).

El Trabajo de Tesis deberá ser redactado en español o inglés.

Art. 44 El Examen de Grado se dará por aprobado si obtiene calificación mayor o igual a 85, en escala de 0 a 100.

Art. 45 Si la calificación del Examen de Grado fuese menor que 85, el Comité de Tesis, dentro de los 5 días hábiles siguientes a la realización del Examen, determinará si conceder o no una última oportunidad para que el Candidato rinda nuevamente el Examen.

TITULO VIII: DEL GRADO ACADÉMICO

Art. 46 Una vez cumplidas por parte del Candidato a Doctor todas las exigencias de Graduación, la Universidad le otorgará el grado académico de "**Doctor en Ingeniería Informática**".

TITULO IX: DE LA RESPONSABILIDAD DEL PRESENTE REGLAMENTO

Art. 47 La responsabilidad de la aplicación de las disposiciones contenidas en el presente reglamento será del Director del DII.

ARTÍCULOS TRANSITORIOS

Art. T1 Los miembros del CDTP que, a la fecha de aprobación de estas normas por parte del CCDP, estén guiando alguna Tesis, podrán continuar haciéndolo con todas las atribuciones y obligaciones asociadas a esta tarea, incluso si no cumplieren los requisitos establecidos en estas normas para conformar el CDTP.

ANEXO Nº 1: ADMINISTRACIÓN DEL PROGRAMA

DIRECCIÓN DEL PROGRAMA

- SOLAR FUENTES, MAURICIO (Director)
- ASTUDILLO ROJAS, HERNÁN (Director Alterno)

COMITÉ DE PROGRAMA (CP) o COMITÉ ACADÉMICO (CADII)

- SOLAR FUENTES, MAURICIO Miembro Titular (Director DII)
- ASTUDILLO ROJAS, HERNÁN Miembro Titular (Director Alterno DII)
- ALLENDE OLIVARES, HÉCTOR Miembro Titular
- CASTRO VALDEBENITO, CARLOS Miembro Titular

CUERPO DE DIRECTORES DE TESIS DEL PROGRAMA (CDTP)

- ALLENDE OLIVARES, HÉCTOR Inteligencia Artificial y Ciencia de Datos
- ARAYA LOPEZ, MAURICIO Computación Aplicada
- ASTUDILLO ROJAS, HERNÁN Software y Ciberseguridad
- CASTRO VALDEBENITO, CARLOS Inteligencia Artificial y Ciencia de Datos
- MENDOZA ROCHA, MARCELO Inteligencia Artificial y Ciencia de Datos
- RIFF ROJAS, MARÍA CRISTINA Inteligencia Artificial y Ciencia de Datos
- ROSAS OLIVO, ERIKA Software y Ciberseguridad
- SALINAS CARRASCO, LUIS Computación Aplicada
- SOLAR FUENTES, MAURICIO Computación Aplicada
- TORRES LOPEZ, CLAUDIO Computación Aplicada

ANEXO Nº 2: NÓMINA DE PROFESORES DEL PROGRAMA DII

Nº	NOMBRE	GRADO ACADÉMICO	ÁREA ESPECIALIZACION
01	ALLENDE OLIVARES, HÉCTOR	Dr. rer. nat. Statistik, Technische Universität Dortmund, Dortmund, Alemania. (1988)	Inteligencia Artificial y Ciencia de Datos
02	ARROYUELO BILLIARDI, DIEGO	Dr. en Ciencias mención Computación, Universidad de Chile, Chile. (2009)	Computación Aplicada
03	ARAYA LOPEZ, MAURICIO	Dr. en Informatique, Université de Lorraine, Nancy, Francia. (2013)	Computación Aplicada
04	ASTUDILLO ROJAS, HERNÁN	Ph.D. Information and Computer Science, Georgia Institute of Technology, Atlanta, GA, EEUU. (1996)	Software y Ciberseguridad
05	BUIL, CARLOS	Ph.D. in Computer Science and Artificial Intelligence, Facultad de Informática, Universidad Politécnica de Madrid, España. (2012).	Inteligencia Artificial y Ciencia de Datos
06	CASTRO VALDEBENITO, CARLOS	Dr. en Informática, INRIA, Université Henri Poincaré, Nancy, Francia. (1998)	Inteligencia Artificial y Ciencia de Datos
07	DOMBROVSKAIA, LIOUBOV	Dr. en Ciencias de la Ingeniería, Pontificia Universidad Católica de Chile, Chile. (1998)	Software y Ciberseguridad
08	LOBOS YÁÑEZ, CLAUDIO	Dr. in Models, Methods and Algorithms in Biology, health and environment. Université Joseph Fourier-Grenoble I, Francia. (2009)	Computación Aplicada
09	LÓPEZ, CLAUDIA	Ph.D. Information Sciences and Technology, University of Pittsburgh, EEUU. (2015).	Software y Ciberseguridad
10	MENDOZA ROCHA, MARCELO	Dr. en Ciencias mención Computación, Universidad de Chile, Chile. (2007)	Inteligencia Artificial y Ciencia de Datos

11	MONGE ANWANDTER, RAÚL	Dr. Ing. Informatik, Universität Erlangen-Nürnberg, Alemania (1992)	Software y Ciberseguridad
12	MOREIRA WENZEL, ANDRÉS	Dr. en Ciencias mención Informática y Matemáticas, Universidad de Chile, Chile. (2003)	Computación Aplicada
13	RIFF ROJAS, MARÍA CRISTINA	Dr. in Mathematics and Informatics. École Nationale de Ponts et Chaussèes, París, Francia. (1997)	Inteligencia Artificial y Ciencia de Datos
14	SALINAS CARRASCO, LUIS	Dr. rer. nat. Mathematik, Universität des Saarlandes, Saarbrücken, Alemania. (1976)	Computación Aplicada
15	SOLAR FUENTES, MAURICIO	Dr. en Ciencias en Ing. de Sistemas y Computación. COPPE, Universidade Federal do Rio de Janeiro, Brasil. (1992)	Computación Aplicada
16	TORRES, CLAUDIO	Ph.D. in Applied Mathematics, University of Delaware, EEUU. (2012).	Computación Aplicada
17	VISCONTI ZAMORA, MARCELLO	Ph.D. in Computer Science, Oregon State University, EEUU. (1993)	Software y Ciberseguridad

ANEXO Nº 3: PLAN DE ESTUDIOS DEL PROGRAMA DII

El estudiante de doctorado debe aprobar un programa de estudios de, al menos, 60 SCT en asignaturas de nivel de postgrado, y una actividad de tesis compuesta por 180 SCT, correspondientes a los Seminarios de Tesis I y II, en un período total de 8 semestres académicos.

<table>
<tr><th colspan="2">AÑO 1</th><th colspan="2">AÑO 2</th><th colspan="2">AÑO 3</th><th colspan="2">AÑO 4</th></tr>
<tr><th>SEM 1</th><th>SEM 2</th><th>SEM 3</th><th>SEM 4</th><th>SEM 5</th><th>SEM 6</th><th>SEM 7</th><th>SEM 8</th></tr>
<tr><td>INF-400</td><td>INF-400</td><td rowspan="4">INF-591 SEMINARIO DE TESIS I</td><td rowspan="4">INF-591 SEMINARIO DE TESIS I</td><td rowspan="4">INF-592 SEMINARIO DE TESIS II</td><td rowspan="4">INF-592 SEMINARIO DE TESIS II</td><td rowspan="4">INF-592 SEMINARIO DE TESIS II</td><td rowspan="4">INF-592 SEMINARIO DE TESIS II</td></tr>
<tr><td>INF-400</td><td>INF-400</td></tr>
<tr><td>INF-500</td><td>INF-500</td></tr>
<tr><td>INF-500</td><td>INF-500</td></tr>
<tr><td>30 SCT</td><td>30 SCT</td><td colspan="2">60 SCT</td><td colspan="4">120 SCT</td></tr>
<tr><td colspan="2">ASIGNATURAS</td><td colspan="6">TRABAJO DE TESIS DOCTORAL</td></tr>
<tr><td colspan="8">240 SCT</td></tr>
<tr><td colspan="8">1 SCT = 27 Horas Cronológicas (Decreto Rectoría 3242020)</td></tr>
</table>

Observaciones:

- El tiempo normal de permanencia en el Programa, para un estudiante regular, es de 4 años.
- La permanencia máxima en el Programa, para un estudiante regular es de 6 años.
- Toda asignatura INF-5XX debe culminar con un informe técnico de investigación.
- Este programa de estudios supone una carga mínima de 60 SCT en el primer año del DII.
- Durante el segundo año, la carga mínima es de 60 SCT, correspondiente al Seminario de Tesis I.
- El Tercer y cuarto año, la carga mínima es de 120 SCT, también correspondiente a 60 SCT anuales (Seminario de Tesis II).
- Se ha omitido el carácter (obligatorio, electivo y/o nivelación) de cada asignatura, ya que es obligación del alumno tomar 4 cursos de nivel 400 y 4 cursos de nivel 500, pero no se especifica cuáles, puesto que dependerá del plan que su tutor y el CADII estimen conveniente para el desarrollo del alumno en concordancia con sus intereses.

Los seminarios INF-591 e INF-592 corresponden a los seminarios de Tesis I y II indicados en el Título V del RI (Art. 32).

Las asignaturas del plan de estudio se clasifican en 3 grupos:

a) Computación Aplicada.
b) Inteligencia Artificial y Ciencia de los Datos.
c) Software y Ciberseguridad.

El plan de estudio del DII contempla 4 asignaturas formativas de nivel 400 de 7 créditos SCT cada una, 4 asignaturas formativas de nivel 500 de 8 créditos SCT cada una, con un total de 60 créditos SCT, más un Seminario de Tesis I con una carga de 60 créditos SCT y finalmente un Seminario de Tesis II con una carga de 120 créditos SCT, lo que da un total de 240 créditos SCT en asignaturas más tesis. La duración total de trabajo efectivo de los estudiantes corresponde a 6.480 horas.

Para fortalecer la preparación de los alumnos del DII, el año 2015 el CADII acordó, conjuntamente con el Comité Académico del Magíster en Ingeniería Informática (CAMII) establecer una base de tres asignaturas obligatorias para todos los alumnos: “Metodología de la Investigación”, “Diseño Avanzado de Algoritmos”, y “Computabilidad y Complejidad Computacional”.

El alumno debe inscribir las asignaturas obligatorias durante el primer año, sin embargo, si lo hace durante el primer o segundo semestre, dependerá del plan de estudio acordado con su tutor. En la malla que se muestra a continuación, se incluyen estas asignaturas a modo de referencia.

Asignaturas nivel 400 y 500.

<table>
<tr><th colspan="2">AÑO 1</th><th colspan="2">AÑO 2</th><th colspan="2">AÑO 3</th><th colspan="2">AÑO 4</th></tr>
<tr><th>SEM 1</th><th>SEM 2</th><th>SEM 3</th><th>SEM 4</th><th>SEM 5</th><th>SEM 6</th><th>SEM 7</th><th>SEM 8</th></tr>
<tr><td>Computabilidad y Complejidad Computacional</td><td>INF-400</td><td rowspan="4">INF-591 Seminario de Tesis I</td><td rowspan="4">INF-591 Seminario de Tesis I</td><td rowspan="4">INF-592 Seminario de Tesis II</td><td rowspan="4">INF-592 Seminario de Tesis II</td><td rowspan="4">INF-592 Seminario de Tesis II</td><td rowspan="4">INF-592 Seminario de Tesis II</td></tr>
<tr><td>INF-400</td><td>INF-400</td></tr>
<tr><td>Diseño Avanzado de Algoritmos</td><td>Metodología de la Investigación</td></tr>
<tr><td>INF-500</td><td>INF-500</td></tr>
<tr><td colspan="2">30 SCT</td><td colspan="2">60 SCT</td><td colspan="4">120 SCT</td></tr>
<tr><td colspan="2">ASIGNATURAS</td><td colspan="6">TRABAJO DE TESIS DOCTORAL</td></tr>
<tr><td colspan="8">240 SCT</td></tr>
<tr><td colspan="8">1 SCT = 27 Horas Cronológicas</td></tr>
</table>

ANEXO Nº 4: ASIGNATURAS DEL PROGRAMA DII

ASIGNATURAS NIVEL 400				
Nombre del curso o seminario	**Académico(s) a cargo**	**Carga horaria total (hr)**	**Créditos totales SCT**	**Período**
Calidad y Productividad de Software	M. Visconti	189	7	Sem 1 o 2
Arquitectura de Software	H. Astudillo	189	7	Sem 1 o 2
Interfaces Hombre-Máquina	L. Dombrovskaia	189	7	Sem 1 o 2
Sistemas Distribuidos	R. Monge	189	7	Sem 1 o 2
Computación Gráfica	C. Lobos	189	7	Sem 1 o 2
Optimización Combinatoria	C. Castro	189	7	Sem 1 o 2
Tópicos Avanzados en Inteligencia Artificial	M.C. Riff	189	7	Sem 1 o 2
Modelamiento Estocástico y Simulación	H. Allende	189	7	Sem 1 o 2
Redes Neuronales Artificiales	H. Allende	189	7	Sem 1 o 2
Reconocimiento de Formas en minería de datos	M. Mendoza	189	7	Sem 1 o 2
Redes Complejas	A. Moreira	189	7	Sem 1 o 2
Tecnologías de Búsqueda Avanzada en la Web	M. Mendoza	189	7	Sem 1 o 2
Web Semántica	C. Buil	189	7	Sem 1 o 2
Elementos de análisis para informática y computación	L. Salinas	189	7	Sem 1 o 2
Fundamentos de inferencia y Aprendizaje	H. Allende	189	7	Sem 1 o 2
Computabilidad y Complejidad Computacional	A. Moreira	189	7	Sem 1 o 2
Tópicos Actuales en Ingeniería de Software	H. Astudillo	189	7	Sem 1 o 2
Tópicos en Sistemas de Información	C. López	189	7	Sem 1 o 2
Introducción a la Computación Cuántica	M. Solar	189	7	Sem 1 o 2
Introducción a la Bioinformática	A. Moreira	189	7	Sem 1 o 2
TEI: Computación Peer to Peer	X. Bonnaire	189	7	Sem 1 o 2
TEI: Seguridad de Sistemas Distribuidos	E. Fernández	189	7	Sem 1 o 2
TEI: Astroinformática	M. Solar	189	7	Sem 1 o 2
TEI: Ingeniería de Software Experimental	M. Visconti	189	7	Sem 1 o 2

Nota:

TEI: Temas Especiales en Informática -asignaturas ofrecidas por los distintos profesores.

ASIGNATURAS NIVEL 500				
Nombre del curso o seminario	**Académico(s) a cargo**	**Carga horaria total (hr)**	**Créditos totales SCT**	**Período**
Métodos Numéricos Computacionales	C. Torres	216	8	Sem 1 o 2
Compresión de Texto	D. Arroyuelo	216	8	Sem 1 o 2
Evaluación de Arquitectura de Software	H. Astudillo	216	8	Sem 1 o 2
Computación Autonómica	H. Astudillo	216	8	Sem 1 o 2
Programación Paralela Aplicada Avanzada	X. Bonnaire	216	8	Sem 1 o 2
Modelos Computacionales en Series de Tiempo	H. Allende	216	8	Sem 1 o 2
Métodos Computacionales en Teoría de Funciones	L. Salinas	216	8	Sem 1 o 2
Diseño Avanzado de Algoritmos	D. Arroyuelo / M. Mendoza	216	8	Sem 1 o 2
Metodología de la Investigación	H. Allende/ H. Astudillo	216	8	Sem 1 o 2
Simulación Mediante Mallas Geométricas	C. Lobos	216	8	Sem 1 o 2
Métodos Cuantitativos en el Procesamiento Computacional de Imágenes	L. Salinas	216	8	Sem 1 o 2
Programación con restricciones	C. Castro	216	8	Sem 1 o 2
Computación Evolutiva	M. C. Riff	216	8	Sem 1 o 2
Lógica Borrosa	C. Moraga	216	8	Sem 1 o 2
Máquinas de Aprendizaje Computacional	H. Allende	216	8	Sem 1 o 2
Sistemas Complejos Discretos	A. Moreira	216	8	Sem 1 o 2
Bases de Datos Documentales	M. Mendoza	216	8	Sem 1 o 2
Seminario de Especialidad I	Director de tesis	216	8	Sem 1 o 2
Seminario de Especialidad II Bioinformática	Director de tesis	216	8	Sem 1 o 2
Seminario de Especialidad II: Cloud Computing y Big Data	Director de tesis	216	8	Sem 1 o 2
Seminario de Especialidad II: Computación Cuántica	Director de tesis	216	8	Sem 1 o 2
Trabajo de tesis				
Seminario de Tesis I (34 créditos)	Director de tesis	1620	60	Sem 1 o 2
Seminario de Tesis II (68 créditos)	Director de tesis	3240	120	Sem 1 o 2

Nota:

Seminario de Especialidad I: Corresponde a temas de especialidad, dictados por el tutor a sus estudiantes.

Seminario de Especialidad II: Corresponde a temas de especialidad, dictados por el tutor a sus estudiantes.

ANEXO Nº 5

Comité Académico del Doctorado en Ingeniería Informática
Departamento de Informática

Pauta para Entrevista a Postulantes al DII

Esta pauta se compone de tres secciones: Pre-entrevista, Entrevista y Post-entrevista. A su vez, la Entrevista se compone de 4 etapas: Introducción, Motivación, Potencial de Inserción, y Financiamiento.

1. Pre-entrevista

El postulante debe contactarse con la Secretaría Técnica del DII para fijar una reunión por video-conferencia (Zoom/Hangouts/Skype/otro), y conocer el material informativo del programa (profesores, áreas, cursos, etc).

2. Entrevista

2.1. Introducción

Dar a conocer al postulante información sobre competencias de perfil de egreso y apoyo DII: el alumno debe investigar y publicar, puede hacer docencia en pregrado y postgrado, puede participar en proyectos I+D+i, debe hablar inglés a la salida del programa, debe tener una pasantía en un centro de excelencia, y participar en conferencias internacionales (SCCC, CLEI, etc).

2.2. Motivación

Se debe verificar con el postulante que entienda la naturaleza científica del programa. El postulante debe entender que no se trata de un postgrado en Tecnología de la Información extendido, como tampoco se trata de hacer una herramienta, etc.
Se debe determinar si el postulante ha realizado investigación y especialmente si ha publicado (aunque sea nacionalmente)
En especial se debe identificar el área de interés en el DII por parte del postulante.

2.3. Potencial de inserción

Se debe estimar posibilidad de convalidación de cursos cuando corresponda.

2.4. Financiamiento

Se debe determinar necesidad de becas y estimar la probabilidad de adjudicarse una beca institucional, nacional (ANID u otra) o dispone de financiamiento propio.

3. Post-entrevista

Se debe redactar un resumen, incluyendo apreciación personal de cada miembro del CADII sobre los ítems 2.2, 2.3 y 2.4. El resumen permite evaluar los criterios de evaluación con las ponderaciones indicadas en la siguiente Tabla.

EXCELENTE – 5 puntos – El/La postulante cumple/aborda de manera sobresaliente todos los aspectos relevantes del criterio en cuestión. Cualquier debilidad es muy menor.
MUY BUENO – 4 puntos – El/La postulante cumple/aborda los aspectos del criterio de muy buena manera, aun cuando son posibles ciertas mejoras.
BUENO – 3 puntos – El/La postulante cumple/aborda los aspectos del criterio de buena manera, aunque requiere ciertas mejoras.
REGULAR – 2 puntos – El/La postulante cumple/aborda en términos generales los aspectos del criterio, pero existen importantes deficiencias.
DEFICIENTE – 1 punto – El/La postulante no cumple/aborda adecuadamente los aspectos del criterio o hay graves deficiencias inherentes.
NO CALIFICA – 0 puntos – El/La postulante no cumple/aborda el criterio bajo análisis o no puede ser evaluada debido a la falta de antecedentes o información incompleta.

CRITERIOS DE EVALUACIÓN Y PONDERACIÓN	SUBCRITERIO ESTADO PARAMETRIZACIÓN	Ponderación
Antecedentes académicos y/o trayectoria y/o experiencia laboral del/de la postulante (60%)	Antecedentes académicos de pregrado	35%
	Actividades de docencia e investigación y antecedentes de postgrados	15%
	Cartas de Recomendación	10%
Objetivos de estudio en que el/la postulante funda su postulación (20%)	Objetivo de estudio	10%
	Declaración de intereses	5%
	Retribución del postulante al país	5%
Nivel, Calidad y Trayectoria del postulante para adjudicarse una beca (institucional, Nacional y/o extranjera) (20%)	Nivel, Calidad y Trayectoria del postulante para adjudicarse una beca de la Dirección de Postgrado (USM), Beca Chile (ANID), o tenga financiamiento propio para realizar sus estudios	20%

ANEXO Nº 6

DECLARACIÓN SIMPLE DEL ALUMNO

En Valparaíso, a ……..., Yo , ……., Cédula de Identidad Nº ……..., de Nacionalidad ……..., con domicilio en **"**……. …….**",** "ciudad", cuya tesis para optar al Grado de Doctor en Ingeniería Informática de la Universidad Técnica Federico Santa María se denomina **"**……..", formulo la siguiente declaración.

DECLARO QUE:

- El contenido de este trabajo de tesis es **original e independiente**, y no ha sido presentado en otra Universidad o Programa de Estudios, tanto a nivel nacional como internacional. Se excluyen de esta situación, los temas de tesis que se hayan entregado en el marco de un convenio actualmente vigente, que mantenga el Programa con otra casa de estudios.
- Dicho tema de tesis, es **exclusivo** para optar al grado de Doctorado en Ingeniería Informática de la Universidad Técnica Federico Santa María.
- Las investigaciones y resultados obtenidos e incluidos en dicho trabajo, son total y verídicamente **reproducibles.**
- Una vez presentado mi trabajo de tesis y firmada esta declaración simple, no podré retractarme, ni solicitar la detención del proceso de evaluación.

Me comprometo, al momento de suscribir esta declaración, a que todo lo señalado anteriormente se ajusta a la realidad.

En el evento que el Comité Académico del Doctorado en Ingeniería Informática estime que el contenido de esta declaración no se ha cumplido fielmente, estoy de acuerdo en que se me aplique la normativa que la Universidad contemple en este caso y que se encuentre actualmente vigente.

---------- ----------

Nombre Alumno Firma

ANEXO Nº 7: ASIGNATURAS DEL PROGRAMA DII

PROGRAMA DE ASIGNATURAS NIVEL 400

Asignatura	Calidad y Productividad de Software	**Sigla**	INF-413
Créditos SCT	7	**Conocimientos previos:** Ingeniería de Software	

Descripción

En este curso se presentan los conceptos clave de medición en Ingeniería de software, particularmente en las áreas de gestión de proyectos y aseguramiento de calidad de software. Se aplican diversas técnicas y modelos del estado del arte para el dimensionamiento de software, la estimación de esfuerzo, tiempos y costos, y la gestión cuantitativa de la calidad en la producción de software. Finalmente, se analizan los fundamentos básicos del mejoramiento de procesos de software.

Objetivos

- Dimensionar software mediante el uso de métricas.
- Aplicar técnicas y modelos de estimación de productividad de software.
- Aplicar técnicas y modelos para la gestión de calidad de software.
- Conocer los fundamentos del mejoramiento de procesos de software

Contenidos

Rol de las mediciones en software: mediciones de producto, proceso, recursos, paradigma Goal Question-Metric.
Métricas de software: clásicas, puntos de función, orientadas a objeto.
Productividad de software: factores, modelos.
Estimación de esfuerzo: modelo Putnam, COCOMO, COCOMO II.
Calidad de software: técnicas de aseguramiento de calidad, mediciones de calidad, gestión cuantitativa de calidad.
Mejoramiento de procesos de software: enfoques, modelos, estado de la práctica.

Metodología

Clases expositivas, presentación y discusión de casos, trabajos prácticos, investigación independiente.

Evaluación

Trabajos prácticos (20%), Trabajo independiente de investigación (20%), Certamen: 20%, Examen final: 40%

Bibliografía

N. Fenton, S.L. Pfleegler, “Software Metrics, A Rigorous and Practical Approach”, IEEE CS, Press, 2da Edición, 1998.
D. Galin, “Software Quality Assurance”, Pearson Education, 2004.
J. Persse, “Process Improvement Essentials”, O´Reilly, 2006.
S. McConnell, “Software Estimation”, Microsoft Press, 2006**.**
[1] IEEE Software.
[2] Software Quality Journal.

Elaborado:	Marcello Visconti	**Observaciones:** Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como curso de especialidad. **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	1/8/2000	

Asignatura	Arquitectura de Software	**Sigla**	INF-414
Créditos SCT	7	**Conocimientos previos:** ILI-225 Ingeniería de Software	

Descripción

El diseño de sistemas de software complejos requiere competencias de concepción, evaluación y construcción diferentes de las requeridas por diseño de aplicativos individuales. El énfasis radica en la satisfacción de propiedades sistémicas ("requisitos extra funcionales") y el uso de tecnologías para sistemas distribuidos. Este curso reporta técnicas, modelos y criterios para describir, evaluar y desarrollar sistemas de software complejos. Los elementos utilizados incluyen ejemplos de documentación de proyectos reales, casos de estudio, talleres grupales de evaluación, y lecturas complementarias.

Objetivos

Al aprobar el curso el alumno podrá:

- comprender arquitecturas de software como guías para la construcción de sistemas
- evaluar, comparar y mejorar especificaciones de arquitectura y las estructuras que ellas denoten, con criterios de calidad intrínseca (técnicos) y extrínsecos (objetivos)
- elaborar y describir arquitecturas de software a partir de especificaciones funcionales y propiedades sistémicas
- distinguir los problemas de arquitectura de los que no lo son, y explicar articuladamente las nociones básicas de la disciplina
- proponer arquitecturas para situaciones concretas

Contenido

1. Arquitectura en el proceso de desarrollo de software.
2. Descripción de arquitecturas: vistas, notaciones, perfiles UML, estilos de arquitectura.
3. Evaluación de arquitecturas: calidad de la especificación, el sistema y el proceso.
4. Reuso de arquitectura: arquitecturas de referencia, componentes, frameworks, patrones de arquitectura, líneas de productos.
5. Recuperación de arquitecturas.
6. Bases tecnológicas: middleware, transparencias de sistemas distribuidos, escalabilidad, disponilibidad.
7. Roles del arquitecto y estructura organizacional.

Metodología

Clases expositivas, proyectos parcelados durante el semestre, informes periódicos sobre lecturas, y presentaciones intermedias de proyectos. Se apunta a conocer el estado del arte en la elaboración, descripción y evaluación de arquitecturas de software, y practicarlo en un contexto de proyectos monitoreados. Se espera participación activa.

Evaluación

Examen escrito

Bibliografía

[1] "Software Architecture in Practice (3rd Ed.)" Len Bass, Paul Clements, Rick Kazman: Addison-Wesley Professional (2012).

[2] "Essential Software Architecture (2nd Ed.)" Ian Gorton: Springer (2011).

[3] "Documenting Software Architectures: Views and Beyond (2nd Ed.)" Paul Clements, Felix Bachmann, Len Bass and David Garlan: Addison-Wesley Professional (2010).

[4] "Evaluating Software Architectures: Methods & Case Studies," Paul Clements, Rick Kazman, Mark Klein: Addison-Wesley (2001).
[5] "Pattern-Oriented Software Architecture, Vol.5: On Patterns and Pattern Languages." Frank Buschmann, Kevlin Henney, Douglas C. Schmidt: Wiley (2007)
[6] "Just Enough Software Architecture: A Risk-Driven Approach". George Fairbanks: Marshall & Brainerd (2010).
[7] Artículos y manuales contingentes al proyecto.
[8] Referencias en Web

Elaborado:	Hernán Astudillo	**Observaciones:** Corresponde a asignatura de la especialidad Desarrollo de Software**.**
Aprobado:	Depto. de Informática	
Fecha:	03/05/2004	**Actualización:** Diciembre 2012

Asignatura	Interfaces Hombre Máquina	**Sigla**	INF-427
Créditos SCT	3	**Conocimientos previos:** Análisis y diseño de sistemas computacionales, ingeniería de software	

Descripción

Este curso trata todos los aspectos relevantes de la compleja interacción entre humanos y computadores. La filosofía de diseño de interfaces centrado en el usuario incluye una vista particular hacia los procesos de análisis de requerimientos, diseño de los prototipos, revisión y mantención de interfaces de sistemas interactivos. Los diferentes estilos de interacción se abarcan desde la perspectiva de proveer guía clara para sus desarrolladores.

Objetivos

Al aprobar el curso el alumno conocerá 451los conceptos generales de la compleja interacción entre personas y computadores, será capaz de diseñar, implantar, administrar, mantener y refinar las interfaces usuarias de sistemas computacionales interactivos. El proyecto realizado a través del semestre le permitirá profundizar sus conocimientos en un área de su interés.

Contenidos

1. Factores Humanos del Software Interactivo.
2. Teorías, Principios y Pautas.
3. Manejo de Procesos de Diseño.
4. Test y Estudio de Facilidad de Uso.
5. Herramientas de Construcción de Interfaces.
6. Manipulación Directa y Ambientes Virtuales.
7. Selección del Menú y Formularios.
8. Tiempo y Velocidad de Respuesta.
9. Presentación: Función y Estilo.
10. Manuales, Ayuda en Línea y Guías de Referencia.
11. Exploración y Visualización de Información.
12. Hypermedia y WWW.
13. Impacto Social e Individual de Interfaces Usuarias.

Metodología

Clases expositivas acompañadas de ejercicios prácticos y realización de un proyecto de investigación sobre un tema particular.

Evaluación

Se realizarán dos certámenes en el semestre (cada uno 25% de la nota final). El proyecto tiene dos entregas: anteproyecto (15 %) y proyecto terminado (35%). La nota final se calcula como el promedio ponderado de las cuatro notas.

Bibliografía

[1] B. Shneiderman, "Designing the User Interface - Strategies for Effective Human-Computer Interaction", Tercera Edición, Addison-Wesley, 1998.

[2] Jacob Nielsen, “Usability Engineering”, Morgan Kauffmann Publishers, 1993.

Elaborado:	Liouba Dombrovskaia	**Observaciones**: Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como curso de especialidad.
Aprobado:	Depto. de Informática	
Fecha:	1/8/2000	

Asignatura	Sistemas Distribuidos	**Sigla**	INF-440
Créditos SCT	7	**Conocimientos previos:** Conceptos de sistemas operativos y redes de computadores, programación de sistemas y lógica matemática.	

Descripción

La asignatura introduce al alumno en los problemas propios de los sistemas distribuidos de computación, donde se estudian técnicas y métodos que pueden ser aplicados en el diseño sistemas y servicios informáticos distribuidos, de manera abordar y resolver satisfactoriamente estos problemas.

Contenido

Caracterización de los sistemas distribuidos. Arquitecturas de Sistemas distribuidos. Programación y comunicación distribuida. Modelos de computación distribuida. Algoritmos distribuidos básicos. Tolerancia a falla y alta disponibilidad. Sistemas de base de datos distribuidos. Transacciones distribuidas. Replicación de datos. Seguridad de información en ambientes distribuidos. Tecnología de middleware

Objetivos

Al finalizar el curso los alumnos serán capaces de:

- Explicar los conceptos fundamentales y paradigmas asociados a computación distribuida.
- Describir los diferentes estilos de arquitecturas de sistemas distribuidos existentes y las técnicas de diseño que se aplican para implementar servicios informáticos distribuidos.
- Entender los algoritmos distribuidos básicos y cómo éstos pueden ser combinados para resolver diferentes tipos de problemas.
- Decidir qué técnicas y métodos son los más adecuados para la resolución de problemas en el área de sistemas distribuidos.

Contenidos

1. Introducción General a los Sistemas Distribuidos (2)
   Motivación. Definición y características de un sistema distribuido. Ventajas y desventajas. Sistemas centralizados vs. distribuidos. Técnicas de distribución. Modelos y arquitecturas distribuidas. Sinopsis del curso.
2. Programación y Comunicación Distribuida (4)
   Paradigmas de programación. Comunicación orientada a mensajes. Sockets en TCP/IP. Multicasting. Protocolos de comunicación de Middleware. Invocación remota de procedimientos. Middleware orientado mensajería (MoM). Data Streaming.
3. Computación Distribuida y Algoritmos Distribuidos (5)
   Modelos de Sistemas Distribuidos. Tiempo, causalidad y ordenamiento de eventos. Relojes Lógicos. Sincronización de relojes. Observaciones válidas. Historias causales y relojes de vector. Cortes, estados globales y consistencia. Algoritmos distribuidos básicos. Elección distribuida. Instantánea distribuida. Detección de término. Detección de deadlock. Exclusión mutua distribuida.
4. Tolerancia a Fallas (4)
   Conceptos y enfoques básicos sobre tolerancia a fallas y alta disponibilidad. Acuerdo bizantino. Sincronización de relojes. Memoria estable. Entrega fiable de mensajes. Broadcast fiable, atómico y causal. Recuperación y estado consistente. Procesos tolerantes a fallas.

5. Base de Datos y Transacciones Distribuidas (5)
   Sistemas de Bases de Datos Distribuidas. Fragmentación y replicación de base de datos. Consultas distribuidas. Transacciones distribuidas. Control de concurrencia y serialización. Recuperación de errores. Protocolos de compromiso. Replicación de datos.
6. Infraestructura y servicios distribuidos (5)
   Servicios de directorio y de nombres. Seguridad computacional e Infraestructura de Clave Pública. Sistemas de Archivos Distribuidos. Tecnología de middleware.

Metodología

Clases lectivas con dos sesiones a la semana para presentación y discusión de la materia, cuya compresión es evaluada en dos partes (2 certámenes).
Un trabajo de investigación donde el alumno aplique los conceptos estudiados en el curso, que al término será presentado a los demás alumnos.

Evaluación

2 Certámenes: 67%

Trabajo de investigación: 33%

Bibliografía

[1] Coulouris, et.al. "*Distributed Systems: Concepts and Design*" 5th. Edition, Addison Wesley, 2011.
[2] A. Tanenbaum, M. van Steen, "Distributed Systems: Principles and Paradigms", 2nd. Edition, Prentice Hall, 2006.
[3] M. Singhla & N.G. Shivaratri, "*Advanced Concepts in Operating Systems*", McGraw-Hill, 1994.

Elaborado:	Raúl Monge	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012.
Fecha:	20/06/2000	

Asignatura	Computación Gráfica	**Sigla**	INF-451
Créditos SCT	7	**Conocimientos previos:** Programación, Estructuras de Datos y Arquitectura de Computadores	

Descripción

En esta asignatura se introduce a la Computación Gráfica. Se define un sistema computacional con gráfica y se programan aplicaciones a través de una librería con funciones gráficas como OpenGL. Temas que se estudian son: Interacción hombre- computador, transformaciones geométricas, proyecciones y modelación. Para estos temas se programan ejemplos en un lenguaje de programación como C/C++.

Objetivos

- Manejar los conceptos fundamentales de la computación gráfica en hardware, software y aplicaciones.
- Aplicar técnicas de programación en C/C++ en aplicaciones gráficas.
- Implantar estructuras de datos para aplicaciones con gráfica.
- Desarrollar programas utilizando las librerías gráficas OpenGL y GLUT.

Contenidos

1. Sistemas Gráficos y Modelos.
2. Programación de Aplicaciones Gráficas.
3. Entrada e Interacción.
4. Objetos Geométricos y Transformaciones.
5. Viewing.
6. Shading.
7. Implementación.
8. Trabajo con Modelos.

Metodología

Cada semana se realiza una clase. Los alumnos deben realizar 2 tareas en el semestre. Además trabaja cada alumno en un tema específico de la computación gráfica lo que implica en el semestre por lo menos 3 presentaciones sobre el tema: Presentación del tema, presentación de avance y presentación final. Al final se entrega un informe escrito. Los trabajos específicos pueden ser estudios bibliográficos o desarrollos de aplicaciones.

Evaluación

Presentaciones 20%
Trabajo Escrito 20%
Tareas 50%
Nota del Profesor 10%

Bibliografía

[1] Edward Angel, Dave Shreiner: "*Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL™*", Addison-Wesley, 6° Ed. 2012 (Texto Guía).
[2] Robert Whitrow: "*OpenGL Graphics through Applications"*, Springer 2008.
[3] Peter Shirley: "*Fundamentals of Computer Graphics"*, A.K. Peters 2° Ed. 2005.
[4] Edward Angel: "*OpenGL: A Primer"*, Addison-Wesley 3° Ed. 2007.
[5] Revista: IEEE Computer Graphics and Applications (ISSN 0272-1716).
[6] Revista: ACM SIGGRAPH Computer Graphics (ISSN 0097-8930).

[7] Revista: Computers & Graphics – An International Journal of Systems & Applications in Computer Graphics (ISSN 0097-8493)

Elaborado: **Aprobado:**	Hubert Hoffmann Depto. de Informática	**Observaciones:** Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como asignatura de especialidad o electiva. **Actualización:** Diciembre 2012
Fecha:	1/8/2000	

Asignatura	Optimización Combinatoria	**Sigla**	INF-472
Créditos SCT	7	**Conocimientos previos:** Programación Lineal	

Descripción

El curso presenta los conceptos fundamentales de la programación entera. Se estudia los modelos clásicos donde la programación entera es aplicada y los fundamentos de los métodos de resolución de los modelos de optimización combinatoria. Los conceptos teóricos son complementados con el uso de herramientas computacionales para el modelamiento y la resolución de problemas de optimización combinatoria.

Objetivos

- Formular modelos matemáticos deterministas para la optimización de operaciones representables por funciones sobre variables discretas.
- Conocer técnicas de resolución de modelos de optimización combinatoria provenientes de la Investigación de Operaciones.

Contenidos

1. Modelos de optimización combinatoria: conceptos básicos, variables enteras, variables binarias, formulación de modelos, modelos clásicos.
2. Resolución de modelos de optimización combinatoria: enumeración exhaustiva, técnicas de ramificación y acotamiento, técnicas de enumeración implícita, planos de corte.
3. Resolución utilizando herramientas computacionales.
4. Lenguajes para la formulación de modelos de optimización combinatoria.

Metodología

La asignatura contempla la realización de clases de exposición de los elementos teóricos, la realización de tareas individuales de aplicación de los conceptos vistos en clases y el estudio de casos y técnicas propuestas en artículos que presentan el desarrollo actual de la materia.

Evaluación

La asignatura se evalúa en base a certámenes (60%), análisis de artículos (20%) y tareas (20%).

Bibliografía

[1] Combinatorial Optimization: Theory and Algorithms, Bernhard Korte & Jens Vygen, Springer, 2012.
[2] Combinatorial Optimization, Rita Malhotra, C. S. Lalitha, Delhi Pankaj Gupta & Aparna Mehra, Narosa, 2006.
[3] A First Course in Combinatorial Optimization, Jon Lee, Cambridge University Press, 2004

Elaborado:	Carlos Castro	**Observaciones:** Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como curso de especialidad. **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	1/8/2000	

Asignatura	Tópicos Avanzados en Inteligencia Artificial	**Sigla**	INF-474
Créditos SCT	7	**Conocimientos previos:** Programación en C, Inteligencia Artificial	

Descripción

Este curso es una revisión de las nuevas tendencias y avances en los métodos de búsqueda basados en heurísticas más conocidos. Tanto los problemas de optimización combinatoria y los problemas de optimización continua serán tratados con mayor énfasis en la combinatoria. Se presentarán las principales técnicas discutiéndolas en forma crítica y sus variaciones. Se utilizarán papers clave incluyendo aplicaciones. Los estudiantes conocerán cómo y por qué estas técnicas funcionan, cuándo aplicarlas, las ventajas respecto de otras técnicas más tradicionales.

Objetivos

- Presentar al estudiante un panorama de las nuevas tendencias y avances en inteligencia artificial.
- Mostrar al alumno el uso de la inteligencia artificial en la resolución de problemas del mundo real.
- Desarrollar una visión crítica respecto a nuevas propuestas en esta área de investigación

Contenidos

1. Repaso de técnicas de inteligencia artificial para la resolución de problemas
2. Estado del arte en problemas clásicos y nuevas aproximaciones para su resolución
3. Formas de evaluar avances en métodos provenientes de la inteligencia artificial
4. Revisión de la literatura reciente proveniente de conferencias y publicaciones en revistas

Metodología

Clases expositivas, seminarios, reuniones y presentación de avances del proyecto

Evaluación

Cada tarea tiene una ponderación de un 10%, revisión de los papers un 10% y el proyecto un 40%.

Bibliografía

[1] Artificial Intelligence: A Modern Approach, D. Russel and P. Norvig, 3rd. Edition, Prentice Hall, 2010.
[2] G. Luger, Artificial Intelligence: Structures and Strategies for Complex Problem Solving, Addison Wesley, 2008.
[3] Reeves Colin, "Modern heuristic technique for combinatorial problems", John Wiley & Sons, 1993.
[4] Artificial Intelligence in the 21st Century, S. Lucci and D. Kopec, Mercury Learning and Information, 2012.
[5] Adicional: Journal of Knowledge Intelligent Systems, Journal of Engineering Applications of Artificial Intelligence, IEEE Computational Intelligence

Elaborado:	Maria Cristina Riff	**Observaciones:** Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como curso de especialidad.
Aprobado:	Depto. de Informática	
Fecha:	1/8/2000	**Actualización:** Diciembre 2012

Asignatura	Modelamiento Estocástico y Simulación		**Sigla**	INF-475
Créditos SCT	7	**Conocimientos previos:** Lenguajes de programación y Estadística computacional		

Descripción

Este curso se introducen los fundamentos de los procesos estocásticos, y sus aplicaciones a la modelización de sistemas complejos. El diseño de un sistema computacional exige no sólo satisfacer ciertas funciones, sino también determinar de forma cuantitativa el comportamiento y la eficacia del mismo. Para ello, típicamente debemos construir modelos que suelen incluir algún elemento estocástico. En el curso se introducen los principales modelos estocásticos, así como su utilidad para la toma de decisiones. El análisis de sistemas complejos suele conducir a problemas de optimización estocásticos de difícil solución. Una alternativa puede ser el empleo de la simulación, que esencialmente, consiste en la experimentación computacional de un modelo estocástico, que describe el comportamiento dinámico del sistema bajo estudio.

Objetivos

- Comprender los fundamentos de los procesos estocásticos y sus aplicaciones.
- Conocer diversos tipos de procesos estocásticos.
- Conocer las diversas etapas en el desarrollo de modelos de simulación.
- Conocer y comprender los principales algoritmos para generar números, variables aleatorias y procesos estocásticos.
- Conocer métodos para efectuar análisis de salidas de un modelo de simulación.
- Conocer métodos de validación de modelos de simulación.

Contenidos

1. *Introducción a los procesos estocásticos:* teorema de Kolmogorov, clasificación de procesos estocáticos , estacionarios y no estacionarios , procesos ergódicos, procesos de Markov, procesos de Poisson, aplicaciones a teoría de colas, sistemas de colas exponencial y no exponencial, y aplicación a modelos de pronósticos.
2. *Análisis Espectral de un proceso estocástico:* función de autocorrelación , función de densidad espectral teorema de Wiener y Kintchine, análisis espectral de procesos simples y bidimencionales, y aplicación a procesamiento de señales.
3. *Introducción a modelos de simulación:* generalidades del modelado, simulación y método de Montecarlo, ejemplos ilustrativos en sistemas de colas.
4. *Generación de números aleatorios:* métodos congruenciales, mixtos, multiplicativos, aditivos y mezclas, métodos de registros desfasados, propiedades Test de aleatoriedad: Test de bondad de ajuste.
5. *Generación de variables aleatorias: m*étodo de la Transformación Inversa, método de la Composición, Técnicas de aceptación y rechazo, generación de variables continuas, discretas, generación de variables correlacionadas, generación de procesos estocásticos.
6. *Análisis de salida de modelos de simulación:* medidas de desempeño, contrastes, intervalos de confianza, métodos de comparación.
7. *Técnicas de reducción de variancia.*
   *Validación de modelos de simulación: v*alidación de datos, validación de supuestos, validación experimental, procedimientos estadísticos de validación

Metodología

Clases expositivas, tareas individuales, presentación de trabajo en forma de seminario

Evaluación

Tareas (30%), Examen (30%) y Proyecto (40%).

Bibliografía

[1] Ross, S.M., "*Simulation*", Ed. Academic Press, 5 Edition, 2012.
[2] Feldman, R.M. and Valdez-Flores, C., "*Applied Probability and Stochastics Process*", Ed. Springer, 2 Edition, 2010.
[3] Nelson, B.L., "*Stochastic Modeling: Analysis and Simulation*", Ed. Dover, 2010.

[4] Stroock, D.W., "*An Introduction to Markov Processes*", Ed. Springer, 2005.
[5] Law, A.M., *"Simulation Modeling and Analysis"*, Ed. McGraw-Hill, 4 Edition, 2006.
[6] Kalos, M.H. and Whitlock, P.A., "Monte Carlo Methods", Ed. Wiley-VCH, 2 Edition, 2008.
[7] Robert, C.P. and Casella, G., "Introducing Monte Carlo Methods with R", Ed. Springer Verlag, 1 Edition, 2009.
[8] Revistas : Simulation, Journal of forecasting, ACM transaction on modeling Computer Simulation, and Statistics and Computing.

Elaborado:	Héctor Allende	**Observaciones:** Abierto a estudiantes de Ingeniería Informática e Ingeniería Civil Informática como curso de especialidad. **Actualización: Diciembre 2012**
Aprobado:	Depto. de Informática	
Fecha:	1/8/2012	

Asignatura	Redes Neuronales Artificiales	**Sigla**	INF-477
Créditos SCT	7	**Conocimientos previos:** Lenguajes de programación, cálculo numérico y estadística computacional	

Descripción

Este curso introduce los fundamentos de las redes neuronales artificiales (*Artificial Neural Network*, ANN) y sus aplicaciones a problemas de Regresión, Clasificación y , Predicción y reconocimiento de patrones en general . En el curso se introducen algunos conceptos básicos de las ANN, para diferentes arquitecturas de redes aplicadas tanto a patrones estáticos como a patrones dinámicos. Se analizan diferentes aplicaciones de las ANN, así como sus posibilidades y limitaciones. Finalmente se estudian las redes neuronales recurrentes analizando los problemas de estabilidad, así como la efectividad y aplicación de algunos algoritmos constructivos.

Objetivos

- Conocer las diferentes arquitecturas de las ANN.
- Conocer diversos tipos de funciones de activación de las ANN.
- Conocer y comprender los principales algoritmos para generar ajustar los pesos de las ANN.
- Conocer los principios básicos de reconocimiento de patrones.
- Conocer los principales avances de las ANN, para los problemas de clasificación y reconocimiento de patrones.

Contenidos

1. *Introducción a las ANN: n*euronas simples y redes, las neuronas como funciones.
2. *Redes de una Capa: f*unción lineal discriminante, separabilidad lineal, discriminante lineal generalizado, técnicas de mínimos cuadrados, perceptrón, discriminante lineal de Fisher.
3. *Redes de Multicapa: r*edes Feed-forward, unidades Threshold y sigmoidal, espacio de pesos simétricos, redes de alto orden, regresión Pursuit, teorema de Kolmogorov.
4. *Redes recurrentes:* la máquina de Boltzman, Back- propagation recurrente, aprendizaje secuencial, aprendizaje reforzado, aplicaciones a modelos de pronósticos.
5. *Funciones de Error: d*iversas funciones de error; Gauss, Minkowski, Varianza, modelos de distribución condicional, problemas de clasificación usando suma de cuadrados, entropía.
6. *Optimización de Parámetros:* diversas técnicas de optimización de parámetros; método del gradiente, gradiente conjugado, método de Newton y método de Levenberg Marquardt.
7. *Tópicos Avanzados en Redes Neuronales:* redes probabilísticas, optimización del aprendizaje, técnicas Bayesianas.
8. *Aplicaciones:* procesamiento de imágenes, selección de características, predicción de series de tiempo.

Metodología

Clases expositivas, tareas individuales, presentación de trabajo en forma de seminario.

Evaluación

Examen (30%), tareas (30%) y presentación de monografía (40%).

Bibliografía

[1] Bishop, *"Neural Networks for Pattern Recognition"*, Ed. Clarendon Press Oxford, 1995.
[2] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[3] C. Sammut, G.I. Web. Encyclopedia of Machine Learning Springer, 2011.
[4] C. Bishop *Pattern recognition and Machine Learning* ", Springer, 2006.
[5] T. Hastie, R. Tibshirani and J. Friedman. The Elements of statistical Learning, Ed Springer 2001.
[6] R.M. Hristev, *"Artificial Neuronal Netwoks"*, Ed.The book ANN,2000.
[7] T.L. Fine, *"Feedforward Neural Network Methodology"*, Ed. Springer Verlag, 1999.
[8] Revistas: IEEEE Trans. on Neuronal Netwoks, Neuronal computing, Pattern Recognition, Journal of Machine Learning, Neuronal computing, Pattern Recognition, Journal of Intelligent Data Analysis, Engineering Optimization.

Elaborado:	Héctor Allende	**Observaciones:** Abierto a estudiantes de Postgrado Ingeniería de otras especialidades. **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	1/8/2012	

Asignatura : Reconocimiento de Formas en Minería de Datos	**Sigla** : INF-479
Créditos SCT : 4	**Prerrequisitos**: Base de Datos Y Estadística Computacional.

Descripción:
Fundamentos teóricos de las técnicas usadas para el reconocimiento de entidades y patrones. Estos fundamentos entregan los conocimientos necesarios para reconocer la aplicación de estas técnicas, así como la base para resolver diversos tipos de problemas de reconocimiento.

Objetivos: Proporcionar al estudiante los fundamentos del Reconocimiento y de sus principales aplicaciones.
Al término del curso el alumno será capaz de:

1. Conocer los principios básicos de reconocimiento de patrones
2. Conocer las diferentes técnicas para reconocimiento de formas.
3. Conocer las principales aplicaciones del reconocimiento de patrones, en Data Mining y Procesamiento de Imágenes.
4. Conocer los principales avances para los problemas de clasificación y reconocimiento de patrones.
5. Conocer los principales avances de las ANN, para los problemas de clasificación y reconocimiento de patrones.

Contenido

1. Introducción al reconocimiento de Patrones: Conceptos básicos Fundamentos, Vectores Aleatorios e Inferencia; análisis discriminate.
2. Aproximación paramétrica al reconocimiento de Patrones: Clasificadores lineales; cuadráticos, Bayesianos otros.
3. Aproximación no paramétrica al reconocimiento de Patrones: función de densidad de Parzen, Vecino más cercano, K- means; LVQ.
4. Métodos de Clasificación no supervisada (Clustering): Clustering parametrico, Clustering no-paramétrico.
5. Redes Neuronales y Clasificación: FANN; Redes Bayesianas,
6. Redes Fuzzy.
7. Métodos de Selección y Extracción de Características

Metodología

Clases expositivas, tareas individuales, presentación de trabajo en forma de seminario.

Evaluación

Examen (40%), tareas y presentación de trabajos (60%).

Bibliografía

[1] R. Duda, P. Hart and D. Stork, “Pattern Classification”, Ed. J. Wiley ,2001.
[2] C. Bishop Pattern recognition and Machine Learning ”, Springer, 2006.
[3] J. Han, M. Kamber, “Data Mining Concepts and Techniques”, Ed. Academic Press, 2001.
[4] Revistas: Pattern Recognition, Pattern Recognition Letters IEEEE Trans. on Neuronal Netwoks, , Statistics and Computing.

Revistas:
IEEEE Trans. on Neuronal Netwoks, Neuronal computing, Pattern Recognition, and Statistics and Computing.

Elaborado :	H. Allende Jul-2002	
Aprobado:	Comité IP del DI Oct/2002	
Actualizado	Jul-2011	

Asignatura	Redes Complejas	**Sigla**	INF-480
Créditos SCT	7	**Conocimientos previos:** Estructuras de Datos, Fundamentos de la Informática, Estadística Computacional.	

Descripción

Este curso es una introducción al reciente campo interdisciplinario de las redes complejas, con énfasis en redes sociales y tecnológicas. Esta área, que trata con grafos de gran tamaño y estructura no aleatoria, ha emergido con fuerza desde fines de los años 90, y ha contribuido a entender una amplia variedad de fenómenos, desde el crecimiento de la Web hasta la propagación de epidemias o rumores, o la formación de comunidades. A la vez ha puesto de manifiesto la existencia de patrones comunes en la conectividad de sistemas tan distintos como la regulación génica, el cerebro humano, las conexiones entre conceptos, o las redes sociales online. El curso cubre los principales modelos, propiedades genéricas, herramientas de análisis y los ejemplos más importantes de redes específicas.

Objetivos

Al aprobar la asignatura el alumno tendrá una perspectiva general del estudio de las redes complejas, su relevancia en distintos ámbitos (con énfasis en las redes más cercanas al quehacer informático), y las herramientas prácticas e intelectuales para su análisis. Además habrá trabajado con datos concretos de redes y estará preparado para abordar proyectos en que este tipo de información sea relevante. En varios temas específicos habrá tenido acceso a literatura de investigación de punta, o estará en condiciones de abordarla.

Contenidos

1. Introducción a las redes de gran tamaño; ejemplos. Modelos clásicos (Erdös-Renyi). Propiedades principales: distancias, clusterización, distribución de grados. Pajek, Network Workbench, Gephi y otras herramientas de análisis y visualización.
2. Modelos principales de redes complejas: Watts-Strogatz, Barabási-Albert. Robustez estructural y dinámica bajo falla o ataque. Otras propiedades topológicas de redes. Otros modelos de redes complejas, estáticos y generativos.
3. Medidas de centralidad en redes: análisis de roles, análisis de enlaces; aplicación en redes sociales y en el grafo de la Web. Asortatividad; “club de ricos”. Identificación de comunidades. Propiedades de navegabilidad y búsqueda descentralizada.
4. Patrones en redes. Introducción a la representación gráfica y a la minería de datos de redes. Muestreo de redes e inferencia de propiedades estructurales. Interacción de características topológicas locales y globales.
5. Redes dinámicas y dinámicas en redes. Ejemplos de estudios en redes sociales, tecnológicas y biológicas. Validación de modelos dinámicos. Fenómenos de difusión y contagio; estrategias de control. Auto-organización y fenómenos emergentes.
6. Aspectos específicos de redes importantes: el grafo de la Web; redes sociales online; internet y otras redes tecnológicas; redes semánticas; colaboración científica; redes biológicas.

Metodología

El curso se desarrollará en clases expositivas de 90 minutos de duración. El alumno deberá elaborar tareas en forma individual para complementar su aprendizaje; principalmente se analizarán datos de redes. Adicionalmente, se asignarán lecturas relevantes a los contenidos del curso, y se evaluarán mediante tareas y/o presentación en clases. El examen final del curso será basado en una prueba escrita.

Evaluación
Tareas y presentaciones 60%
Interrogaciones y examen 40%

Bibliografía

[1] Brandes, U., Erlebach, T. (Eds.), *Network Analysis: Methodological Foundations* (LNCS 3418), Springer-Verlag, 2005.
[2] Bornholdt, S., Schuster, H.G. (Eds.), *Handbook of Graphs and Networks: From the Genome to the Internet.* Wiley-VCH, 2003.
[3] Caldarelli, G.,Vespignani, A. (Eds.), *Large Scale Structure And Dynamics Of Complex Networks*. Word Scientific, 2007.
[4] Cook, D., Holder, L. (Eds.), *Mining Graph Data*. John Wiley & Sons, 2006.
[5] Barrat, A., Barthélemy, M., Vespignani, A., *Dynamical Processes on Complex Networks*. Cambridge University Press, 2008.
[6] Newman, M., Networks: An Introduction. Oxford University Press, 2010.

Elaborado:	A. Moreira	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	01-07-2010	

Asignatura	Tecnologías de búsqueda en la Web	**Sigla**	INF-481
Créditos SCT	7	**Conocimientos previos:** estructuras de datos y algoritmos, bases de datos, probabilidad y estadística.	
Hrs. Cat. Sem. : 3	Hrs. Ayud. Sem. : 0	Hrs. Lab. Sem. : 0	

Descripción
La asignatura cubre los fundamentos de recuperación de información que permiten comprender y desarrollar tecnologías de búsqueda de información en la Web. Cubre también temas más avanzados que permiten que el estudiante comprenda el estado del arte de esta área, entre ellas búsqueda en la Web social, búsqueda vertical y búsqueda por facetas.

Objetivos
Al aprobar la asignatura el estudiante será capaz de:
1. Comprender los fundamentos de recuperación de información en texto.
2. Aplicar modelos de recuperación de información en sistemas de búsqueda en la Web.
3. Comprender los principales desafíos y limitaciones de los métodos existentes para realizar búsquedas en la Web social.
4. Comprender y conocer tendencias en búsqueda en la Web.
5. Analizar nuevas tecnologías y métodos de búsqueda, entre ellas búsqueda en la Web social, búsqueda vertical y búsqueda por facetas.

Competencias transversales
1. Comunicar información oral y escrita de manera eficaz.
2. Actuar con autonomía, flexibilidad, iniciativa, y pensamiento crítico al enfrentar problemas nuevos.

Competencias especificas
1. Comprender y evaluar nuevas tecnologías de búsqueda en la Web.
2. Implementar nuevos algoritmos de búsqueda para la Web 1.0 y 2.0.

Contenidos
[1]Fundamentos de recuperación de información en texto: Leyes del texto, técnicas de extracción de información desde texto, estructuras de datos para texto, evaluación de recuperación de información en texto.
[2]Modelos de recuperación de información en texto: Modelo Booleano, modelo Tf-Idf, modelo probabilístico, modelos con retroalimentación de usuarios.
[3]Búsqueda en la Web: Caracterización de la Web, extracción de información en la Web (*crawling*), análisis de hiper-enlaces, arquitecturas de motores de búsqueda.
[4]Búsqueda en la Web 2.0: Caracterización de la Web 2.0, análisis de redes sociales en la Web.
[5]Temas avanzados: Búsqueda vertical, búsqueda por facetas, minería de opiniones, categorización de texto.

Metodología
1. Clases expositivas con apoyo de medios audiovisuales.
2. Desarrollo de ejercicios en clases que permitirán ilustrar los conceptos del área.
3. Actividades grupales: Tarea de procesamiento de texto en grupo y proyecto a desarrollar en el semestre que permitirá trabajar en torno al desarrollo de un sistema de búsqueda en la Web.

4. Actividades individuales: Presentación de un tema de investigación en clases, desarrollo de un trabajo escrito sobre un tema avanzado del estado del arte.

Evaluación

- 3 certámenes
- 1 Proyecto
- 1 Tarea
- 1 Presentación en clases
- 1 Informe escrito sobre tema avanzado

Calificación

- Promedio certámenes: 40%
- Nota proyecto: 20%
- Nota tarea: 10%
- Nota presentación: 10%
- Nota informe escrito: 20%

Bibliografía

1. Büttcher, S., Clarke, Ch. & Cormack, G. "Information Retrieval: Implementing and Evaluating Search Engines", MIT Press, 2010.
2. Manning, Ch., Raghavan, P., Schutze, H. "Introduction to Information Retrieval"
, Cambridge University Press, 2008.

Revistas

Journal of Information Retrieval (JIR), Journal of the American Society for Information Systems and Technology (JASIST), ACM Transactions on the Web (TWEB), ACM Transactions on Information Systems (TOIS), ACM Transactions on Intelligent Systems and Technology (TIST), Journal of Web Engineering (JWE).

Elaborado:	Marcelo Mendoza	**Observaciones:** Actualizado Diciembre 2012
Aprobado:	Depto. Informática 07-07-2011	
Fecha:	24-06-2011	

Asignatura	Web Semántica	**Sigla**	INF-484
Créditos SCT	6	**Conocimientos previos:** Base de Datos.	
Hrs. Cat. Sem. : 3	Hrs. Ayud. Sem. : 0	Hrs. Lab. Sem. : 0	

Descripción

Al final de esta asignatura, el estudiante será capaz de utilizar tecnologías de la Web Semántica para lograr ese objetivo, logrando que sus propias aplicaciones puedan interpretar automáticamente el contenido de una página web, hacer que sus aplicaciones se entiendan entre ellas o que esas mismas aplicaciones razonen automáticamente acerca de los datos que ellas generan. Además, el estudiante será capaz de desarrollar páginas web cuyos contenidos sean entendidos por los grandes jugadores de Internet: Google, Bing, Yahoo y Yandex.

Objetivos

- Desarrollar, implantar y mantener sistemas de software confiables, eficientes y factibles.
- Diseña modelos de datos de la Web Semántica, utilizando un framework reconocido universalmente.
- Diseña modelos de dominio para la Web Semántica, utilizando lenguajes estándares en el área.
- Desarrolla una Web Semántica, aplicando las tecnologías vistas en clases

Contenidos

El curso está dividido en los siguientes módulos, en los cuales se estudian las distintas tecnologías que componen la llamada Web Semántica.

1. **La Web actual**

En este módulo se mostrarán brevemente los problemas de la Web actual y se introducirán las tecnologías de la Web Semántica. Todas estas tecnologías son estándares del World Wide Web Consortium (W3C1).

2. **El modelo de datos de la Web Semántica**

En este módulo se verá el modelo de datos de la Web Semántica. Este modelo es el Resource Description Framework (RDF) [GKM], el cual se basa en describir los datos como relaciones de tres elementos (sujeto, predicado y objeto). Todos los datos de la Web Semántica utilizan este modelo de datos

3. **Modelado de dominios de datos**

Para modelar los datos de un dominio concreto la Web Semántica dispone de dos lenguajes: RDF Schema y Web Ontology Language (OWL) [DS04, Gro12]. Mediante estos dos lenguajes es posible definir restricciones sobre los datos con diverso nivel de detalle. Se aprenderá desde cómo definir clases de datos a definir restricciones universales o existenciales sobre eso mismos datos, además de entender la complejidad asociada a esas restricciones.

4. **El lenguaje de consulta de la Web Semántica**

Una vez comprendidos cómo se estructuran los datos es el momento de acceder a ellos. Para acceder a esos datos se utilizará el lenguaje de consulta SPARQL [PS04, HS10]. Se verán los elementos principales del lenguaje y entenderá la complejidad del mismo.

5. **Integración de datos**

Integración de datos [Len02] es combinar datos de varias fuentes de tal forma que parezca que ellos provienen de una misma fuente. Las tecnologías de la Web Semántica están diseñadas para que el proceso de integración de datos provenientes de diversas fuentes sea lo más sencillo posible. Se verá el conjunto de tecnologías Relational Database 2 RDF (RDB2RDF2) para acceder a datos almacenados en bases de datos relacionales y posteriormente convertir esos datos al modelo RDF.

6. **Aplicaciones de la Web Semántica**.

En este módulo los alumnos verán dos aplicaciones que utilizan las tecnologías vistas durante el curso. Estas aplicaciones son Bio2RDF [CCAD13], la cual integra datos provenientes de más de 40 fuentes en el ámbito de la biomedicina y el proyecto de la Biblioteca del Congreso Nacional de Chile [CSSLG11] el cual modela las normas y leyes chilenas utilizando RDF, RDF Schema y OWL y provee acceso a esas normas a través de SPARQL.

Metodología

En este curso se trabaja con el formato de curso conocido como Blended Learning [GK04]. El curso estará dividido en dos partes fundamentales: en la primera parte se utilizarán clases no presenciales a través de vídeos en la plataforma Coursera (dictados por el actual profesor), complementadas con discusiones presenciales (en clase) acerca de los conceptos y problemas mostrados en estos vídeos. Es decir, los alumnos deberán ver los videos de la plataforma Coursera en casa y después en clase presencial se discutirán los conceptos aprendidos. El profesor se pondrá de acuerdo con los alumnos al inicio del curso. En la segunda parte del curso, los alumnos propondrán soluciones a los problemas vistos anteriormente o bien posibles aplicaciones de las tecnologías anteriores en el dominio que el alumno proponga. Dichas propuestas se implementarán y formarán parte del proyecto final del curso. Este proyecto será individual.

Evaluación

- Proyecto web del curso: 15%
- Certamen y lecturas: 35%
- Proyecto (desarrollo): 25%
- Proyecto (presentación): 25%

Bibliografía

Básica

- Grigoris Antoniou and Frank van Harmelen. A semantic web primer. MIT Press, 2004.
- Pascal Hitzler, Markus Krotzsch, and Sebastian Rudolph. Foundations of semantic web technologies. CRC Press, 2009

Recomendada

AGP09] Marcelo Arenas, Claudio Gutierrez, and Jorge Pérez. Reasoning web. Semantic technologies for information systems. Foundations of RDF Databases, pages 158–204. Springer-Verlag, Berlin, Heidelberg, 2009.
[AvH04] Grigoris Antoniou and Frank van Harmelen. A semantic web primer. MIT Press, 2004.
[CCAD13] Alison Callahan, Jose Cruz-Toledo, Peter Ansell, and Michel Dumontier. Bio2rdf release 2: Improved coverage, interoperability and provenance of life science linked data. In The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings, pages 200–212, 2013.
[CSSLG11] Francisco Cifuentes-Silva, Christian Sifaqui, and Jose Emilio Labra-Gayo. Towards an architecture and adoption process for linked data technologies in open government contexts: A case study for the library of congress of Chile. In Proceedings of the 7th International Conference on Semantic Systems, I-Semantics '11, pages 79–86, New York, NY, USA, 2011. ACM.
[DS04] Mike Dean and Guus Schreiber. OWL Web Ontology Language Reference, 2004.
[GHMP11] Claudio Gutierrez, Carlos A. Hurtado, Alberto O. Mendelzon, and Jorge Pérez. Foundations of semantic web databases. J. Comput. Syst. Sci., 77(3):520–541, May 2011.
[GK04] D Randy Garrison and Heather Kanuka. Blended learning: Uncovering its transformative potential in higher education. The internet and higher education, 7(2):95–105, 2004.
[GKM] J. J. Carroll G. Klyne and B. McBride. Resource description framework (RDF): Concepts and abstract syntax.

[Gro12] W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview (Second Edition), 2012.
[HKR09] Pascal Hitzler, Markus Krotzsch, and Sebastian Rudolph. Foundations of semantic web technologies. CRC Press, 2009.
[HS10] S. Harris and A. Seaborne. Sparql 1.1 query language, 2010.
[Len02] M. Lenzerini. Data integration: A theoretical perspective. In PODS, pages 233–246, 2002.
[MAHP11] Alejandro Mallea, Marcelo Arenas, Aidan Hogan, and Axel Polleres. On blank nodes. In Proceedings of the 10th International Conference on The Semantic Web -Volume Part I, ISWC'11, pages 421–437, Berlin, Heidelberg, 2011. Springer-Verlag.
[PAG09] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. TODS, 34(3), 2009.
[PS04] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF, January 2004

Elaborado:	Carlos Buil	**Observaciones:**
Aprobado:	Depto. Informática	
Fecha:		

Asignatura	Elementos de Análisis para Informática y Computación	**Sigla**	INF-485
Créditos SCT	4	**Conocimientos previos:** Autorización del Profesor con base en una entrevista personal con el estudiante.	

Descripción

Asignatura centrada en los conceptos, resultados y métodos del Análisis necesarios para abordar los temas de la investigación avanzada propios de la Informática y la Ciencia de la Computación contemporáneas.

Objetivos

Capacitar al estudiante en los temas fundamentales del Análisis que aparecen en la investigación avanzada y las publicaciones recientes en la Informática y la Ciencia de la Computación contemporáneas.

Contenidos

1. Elementos del análisis matemático.
2. Elementos del análisis funcional. Elementos de la teoría de operadores.
3. Elementos de la teoría de los “kernels” y de los “kernels” reproductores. Elementos de la teoría de Nachman Aronszajn y Stefan Bergman.
4. Elementos de la teoría de probabilidades e integración. Teoría de la información y entropía.

Metodología

Lecciones del Profesor, seguidas a lo largo del semestre, por una serie de exposiciones de los estudiantes. Trabajo tutorial del Profesor con cada estudiante para preparar las exposiciones de éstos. Discusión crítica de los temas tratados, con miras a proyectar los como temas de investigación personal. Estudio y trabajo personal (textos y artículos).

Evaluación

Mediante tareas periódicas y exposiciones de artículos relevantes en la temática de la asignatura, por parte de los alumnos, y de las subsecuentes discusiones técnicas. Una parte importante de las tareas será de naturaleza experimental (computador). La calificación final de cada estudiante será el promedio aritmético de todas sus evaluaciones parciales.

Bibliografía

[1] Walter Rudin. Real and Complex Analysis. McGraw-Hill Education; 3d edition (May 1, 1986).
[2] Walter Rudin. Functional Analysis. McGraw-Hill Science/Engineering/Math; 2 edition (January 1, 1991).
[3] John B. Conway. A Course in Functional Analysis. Springer Science & Business Media, Sep 7, 1990 - Mathematics - 399 pages.
[4] Nachman Aronszajn, (1950). Theory of Reproducing Kernels. Transactions of the American Mathematical Society 68 (3): 337–404. doi:10.1090/S0002-9947-1950-0051437-7.JSTOR 1990404. MR 51437.
[5] Alain Berlinet and Christine Thomas. Reproducing kernel Hilbert spaces in Probability and Statistics. Kluwer Academic Publishers, 2004.
[6] Cucker, Felipe; Smale, Steve (2002). On the Mathematical Foundations of Learning. Bulletin of the American Mathematical Society 39 (1): 1–49. doi:10.1090/S0273-0979-01-00923-5.MR 1864085.
[7] J. Weickert, H. Hagen (Eds.). Visualization and Processing of Tensor Fields. Springer, Berlin, 2006.
[8] Thomas M. Cover, Joy A. Thomas. Elements of Information Theory. Wiley, 1991.
[9] Bernhard Schölkopf, Alexander J. Smola. Learning with Kernels. The MIT Press. Cambridge, Massachusetts, 2002

Elaborado:	Luís Salinas	**Observaciones:**
Aprobado:	Depto. de Informática	
Fecha:	09/08/2015	

Asignatura	Fundamentos de Inferencia y Aprendizaje	**Sigla**	INF-486
Créditos SCT	4	**Conocimientos previos**: Estadística Computacional Computación Científica	

Descripción

Esta asignatura forma parte del núcleo de cursos fundamentales del plan de estudios del postgrado en Ingeniería Informática y cubre fundamentos teóricos de las máquinas de Inferencia. Incluye tópicos básicos de probabilidades, inferencia estadística y aprendizaje, modelos estadísticos y métodos. Ella es una excelente preparación para el trabajo avanzado en Ciencias de la Computación, Inteligencia Artificial, Aprendizaje Automático, métodos cuantitativos en seguridad informática, Ingeniería de software entre otros.

Objetivos

Capacitar al estudiante en los temas fundamentales del Análisis y la Estadística que aparecen en la investigación avanzada y las publicaciones recientes en la Informática y la Ciencia de la Computación contemporáneas.

Contenidos

Parte I: Breve revisión de teoría de probabilidades

1.1 Probabilidad, medidas de probabilidad, independencia, probabilidad condicional, teoremas.
1.2 Variable aleatoria, función de distribución, Distribuciones Multivariadas, Esperanza, Varianza y Covarianza, Transformaciones, Inecuaciones de Markov, Chebychev y Hoeffding´s.
1.3 Convergencia de variables aleatorias, tipos de convergencia, ley débil de los grandes números, teorema central del límite, método delta.

Parte II: Inferencia Estadística y Aprendizaje

2.1 Modelos paramétricos y no paramétricos.
2.2 Estimación Puntual, estimación por regiones y de distribuciones de probabilidad
2.3 El método Bootstrap, intervalos de confianza Bootstrap.
2.4 Inferencia paramétrica: suficiencia, reducción y Verosimilitud teoremas.
2.5 Contraste de hipótesis y el valor p: Lema de Neyman Pearson, comparación de Algoritmos.
2.6 Contraste de permutaciones, comparaciones Múltiples, Método de Bonferroni.
2.7 Inferencia Bayesiana: El método de Bayes, Funciones de parámetros, simulación, Estimación multi-paramétrica.
2.8 Teoría estadística de decisiones: Estimación del riesgo, funciones de pérdida, máxima verosimilud, minimax y Bayes.

Parte III: Aplicaciones

Nota: Dado la extensión se sugiere seleccionar un tema para su formulación desde la perspectiva estadística.
3.1 Clasificación.
3.2 Modelos lineales y log-lineales
3.3 Inferencia causal.
3.4 Modelos de predicción.
3.5 Clustering.

Metodología
Lecciones de los Profesores, seguidas a lo largo del semestre, por una serie de exposiciones de los estudiantes. Trabajo tutorial de los Profesores con cada estudiante para preparar las exposiciones de éstos. Estudio y trabajo personal (textos y artículos).

Evaluación
Promedio de Tareas 40%
Presentaciones 30%
Examen 30%

Bibliografía

- [1] Wasserman, L. (2004). All of Statistics: A concise course in statistical inference. Ed. Springer Verlag.
- [2] Casella, G. and Berger, R. L. (2002). Statistical Inference, 2nd. Ed Cengage Learning.
- [3] Bickel, P. J. and Doksum, K. A. (2015).Mathematical Statistics. 2nd. Ed CRC Press.
- [4] Rice, J. A. (2010). Mathematical Statistics and Data Analysis, 3era Ed. Thomson.

Elaborado:	H. Allende	**Observaciones**:
Aprobado:	Marzo 2016	
Fecha:		

Asignatura	Computabilidad y Complejidad Computacional	**Sigla**	INF-487
Créditos SCT	4	**Conocimientos previos**:	

Descripción
Este curso cubre los fundamentos de las teorías de computabilidad y complejidad computacional, bases de la informática teórica. En el caso de la computabilidad lo que se estudia son los límites de lo que un computador es, en principio, capaz de computar. Para esto se estudian las nociones de lenguajes formales, gramáticas y máquinas abstractas, con énfasis en el modelo de la máquina de Turing, su universalidad de cómputo y la existencia de problemas indecidibles. En el caso de la complejidad computacional lo que se estudian son los límites de lo que, para efectos prácticos, podemos calcular en un computador. Para esto se definen las clases más importantes de problemas (con énfasis en P y NP), la jerarquía de clases en que se insertan, sus problemas prototípicos, y las reducciones que permiten clasificar a otros problemas y con ello tener cotas para su dificultad.

Objetivos
Al aprobar la asignatura el alumno conocerá las nociones fundamentales de la computabilidad y la complejidad computacional, entendiendo la frontera entre lo indecidible y lo decidible, y dentro de esto último, la frontera entre lo tratable y lo intratable. Sabrá identificar cuando un problema tiene solución algorítmica, y demostrarlo. También sabrá identificar y demostrar cuando un problema tiene solución eficiente, y demostrarlo. Conocerá los problemas prototípicos de las principales clases de complejidad, y sabrá reducir problemas para clasificar su dificultad en función del tiempo y espacio que su resolución requiere.

Contenidos
1. Máquinas de Turing. Lenguajes recursivos y recursivos enumerables; funciones computables.
2. Máquina de Turing universal. Tesis de Church-Turing.
3. Problemas indecidibles, teorema de Rice y funciones no computables.
4. Complejidad algorítmica.
5. Clases de complejidad computacional. Reducciones eficientes y completitud.
6. P y NP. C-Value y C-Sat; teorema de Cook-Levin. Otros problemas NP-completos. CoNP, EXP, NEXP.
7. Diagonalización y teoremas de jerarquía. Oráculos.
8. Complejidad espacial. PSPACE y NL.
9. Complementos: definición de otras clases importantes (BPP, NC, BQP); dificultad de aproximación y teorema PCP.

Metodología
El curso se desarrollará en clases expositivas de 90 minutos de duración. El alumno deberá elaborar tareas en forma individual para complementar su aprendizaje. Adicionalmente habrá interrogaciones y evaluaciones escritas.

Evaluación
Tareas 50%
Interrogaciones y examen 50%

Bibliografía
1. Arora & Barak, Computational Complexity: A Modern Approach. CUP, 2009.
2. Sipser, Introduction to the Theory of Computation. Course Technology, 2005.
3. Moore & Mertens, The Nature of Computation. OUP, 2011.

4. Papadimitriou, Computational Complexity. Addison Wesley, 1994

Elaborado:	A. Moreira	Observaciones: Asignatura Obligatoria para el Programa Magíster en Ciencias de la Ingeniería Informática - Asignatura Obligatoria para el Programa Doctorado en Ingeniería Informática
Aprobado:		
Fecha:		

Asignatura	Tópicos Actuales en Ingeniería de Software	**Sigla**	INF-488
Créditos SCT	4	**Conocimientos previos**:	

Descripción

La Ingeniería de Software es la disciplina que se ocupa de la construcción sistemática, eficaz y eficiente de sistemas de software. Casi toda la Ciencia de Computación existe sobre la programación, pero el desarrollo de sistemas que satisfagan expectativas conflictivas y ambiguas de terceros es mucho más complejo, y la creación y mantención de equipos y organizaciones que hagan esto sistemáticamente es el foco de esta disciplina. Los participantes serán expuestos a las corrientes históricamente principales de investigación de la comunidad, y un panorama de la investigación actual que incluye subdisciplinas, temas transversales, y enfoques inter-disciplinarios.

Objetivos

Comprender tendencias de investigación en Ingeniería de Software. Analizar y redactar reviews sistemáticos del estado del arte de Ingeniería de Software.

Contenidos

1. La comunidad de investigación en Ingeniería de Software

Motivaciones y temas iniciales; miembros destacados; hitos en la literatura; foros, conferencias y revistas "core".

2. Sub-disciplinas y temas recurrentes

La comunidad de Ing.Sw se divide en sub-comunidades, interesadas en problemas específicos y con criterios de validez para problemas y soluciones propuestos. El curso explorará:

a) Ingeniería de Requisitos: funcional/escenarios/goal-oriented
b) Diseño y Arquitectura de Software: ADLs, Architectural Knowledge
c) Sub-comunidades específicas a atributos de calidad: Seguridad, Escalabilidad, otras.
d) Desarrollo de Software: procesos/prácticas, agilismo, DevOps
e) Quality Assurance (QA): testing
f) Software Process Improvement (SPI): modelos de madurez, gestión de conocimiento
g) Ecosistemas de Software.

3. Temas transversales a sub-disciplinas

Existen varios temas que son abordados en varias sub-disciplinas:

a. Variabilidad: incluyendo Model-Driven Engineering
b. Rationale y trazabilidad: incluyendo Design Rationales

4. Enfoques inter-disciplinarios

a. Search-based Software Engineering (SBSE): aplicaciones de técnicas de Inteligencia Computacional
b. Empirical Software Engineering (ESE): aplicaciones para generación de conocimiento confiable
c. Ontologías, Social Network Analysis, y otras técnicas que tratan a la Ing.Sw misma como objeto de conocimiento.

Metodología

El curso consistirá en:

a. Clases lectivas normales, con abundante material complementario previo. controles periódicos de lectura, e informes de lo tratado en cada clase.
b. Desarrollo de ensayos temáticos por los participantes.
c. Charlas de investigadores visitantes, reconocidos en la disciplina.

Evaluación

a. Controles (de lectura y contenido) 40%

b. Ensayos temáticos 30%
c. Informes de clases/charlas 10%
d. Propuesta (no accionable) de posible investigación 20%

Bibliografía
Básica.

• "Towards general theories of software engineering". Special issue of Science of Computer Programming. Pontus Johnson, Mathias Ekstedt, Michael Goedicke, Ivar Jacobson. Science of Computer Programming, 101 (2015) pp 1–5. DOI: 10.1016/j.scico.2014.11.005
• SWEBOK v3.0 - Guide to the Software Engineering Body of Knowledge. Pierre Bourque and Richard Fairley (Eds.). IEEE Computer Society (2014). ISBN: 978-0-7695-5166-1.
• Perspectives on the Future of Software Engineering: Essays in Honor of Dieter Rombach. Jürgen Münch and Klaus Schmid (Eds.). Springer-Verlag Berlin Heidelberg (2013). DOI: 10.1007/978-3-642-37395-4

Recomendada.

- Actas de ICSE (International Conference in Software Engineering), Actas de ASE (I.C. on Automated Software Engineering), RE (I.C. on Requirements Engineering), ECSA (Euro.C. on Software Architecture), EuroSPI (Euro.C. on Software Process Improvement), y otras conferencias temáticas de similar categoría
- Revistas top: IEEE Transactions in Software Engineering (TSE), ACM Transactions on Software Engineering and Methodology (TOSEM), IEEE Software, ACM Sigsoft Notes on Software Engineering (SEN)
- Revistas específicas: Journal of Requirements Engineering, Journal of Systems and Software, Science of Computer Programming, otras
- Revistas generales: IEEE Computer, Communications de la ACM, IT Pro

Elaborado:	H. Astudillo	**Observaciones**:
Aprobado:		
Fecha:		

Asignatura	Tópicos en Sistemas de Información	**Sigla**	INF-489
Créditos SCT	4	**Conocimientos previos**:	

Descripción

La Ingeniería de Software es la disciplina que se ocupa de la construcción sistemática, eficaz y eficiente de sistemas de software. Casi toda la Ciencia de Computación existe sobre la programación, pero el desarrollo de sistemas que satisfagan expectativas conflictivas y ambiguas de terceros es mucho más complejo, y la creación y mantención de equipos y organizaciones que hagan esto sistemáticamente es el foco de esta disciplina. Los participantes serán expuestos a las corrientes históricamente principales de investigación de la comunidad, y un panorama de la investigación actual que incluye subdisciplinas, temas transversales, y enfoques inter-disciplinarios.

Objetivos

Comprender tendencias de investigación en Ingeniería de Software. Analizar y redactar reviews sistemáticos del estado del arte de Ingeniería de Software.

Contenidos

1. La comunidad de investigación en Ingeniería de Software

Motivaciones y temas iniciales; miembros destacados; hitos en la literatura; foros, conferencias y revistas "core".

2. Sub-disciplinas y temas recurrentes

La comunidad de Ing.Sw se divide en sub-comunidades, interesadas en problemas específicos y con criterios de validez para problemas y soluciones propuestos. El curso explorará:

1. Ingeniería de Requisitos: funcional/escenarios/goal-oriented
2. Diseño y Arquitectura de Software: ADLs, Architectural Knowledge
3. Sub-comunidades específicas a atributos de calidad: Seguridad, Escalabilidad, otras.
4. Desarrollo de Software: procesos/prácticas, agilismo, DevOps
5. Quality Assurance (QA): testing
6. Software Process Improvement (SPI): modelos de madurez, gestión de conocimiento
7. Ecosistemas de Software.

3. Temas transversales a sub-disciplinas

Existen varios temas que son abordados en varias sub-disciplinas:

a. Variabilidad: incluyendo Model-Driven Engineering
b. Rationale y trazabilidad: incluyendo Design Rationales

4. Enfoques inter-disciplinarios

a. Search-based Software Engineering (SBSE): aplicaciones de técnicas de Inteligencia Computacional
b. Empirical Software Engineering (ESE): aplicaciones para generación de conocimiento confiable
c. Ontologías, Social Network Analysis, y otras técnicas que tratan a la Ing.Sw misma como objeto de conocimiento.

Metodología

El curso consistirá en:

a. Clases lectivas normales, con abundante material complementario previo. controles periódicos de lectura, e informes de lo tratado en cada clase.
b. Desarrollo de ensayos temáticos por los participantes.
c. Charlas de investigadores visitantes, reconocidos en la disciplina.

Evaluación

a. Controles (de lectura y contenido) 40%

b. Ensayos temáticos 30%
c. Informes de clases/charlas 10%
d. Propuesta (no accionable) de posible investigación 20%

Bibliografía
Básica.

• "Towards general theories of software engineering". Special issue of Science of Computer Programming. Pontus Johnson, Mathias Ekstedt, Michael Goedicke, Ivar Jacobson. Science of Computer Programming, 101 (2015) pp 1–5. DOI: 10.1016/j.scico.2014.11.005
• SWEBOK v3.0 - Guide to the Software Engineering Body of Knowledge. Pierre Bourque and Richard Fairley (Eds.). IEEE Computer Society (2014). ISBN: 978-0-7695-5166-1.
• Perspectives on the Future of Software Engineering: Essays in Honor of Dieter Rombach. Jürgen Münch and Klaus Schmid (Eds.). Springer-Verlag Berlin Heidelberg (2013). DOI: 10.1007/978-3-642-37395-4

Recomendada.

- Actas de ICSE (International Conference in Software Engineering), Actas de ASE (I.C. on Automated Software Engineering), RE (I.C. on Requirements Engineering), ECSA (Euro.C. on Software Architecture), EuroSPI (Euro.C. on Software Process Improvement), y otras conferencias temáticas de similar categoría
- Revistas top: IEEE Transactions in Software Engineering (TSE), ACM Transactions on Software Engineering and Methodology (TOSEM), IEEE Software, ACM Sigsoft Notes on Software Engineering (SEN)
- Revistas específicas: Journal of Requirements Engineering, Journal of Systems and Software, Science of Computer Programming, otras
- Revistas generales: IEEE Computer, Communications de la ACM, IT Pro

Elaborado:	H. Astudillo	**Observaciones**:
Aprobado:		
Fecha:		

Asignatura	**Tema Especial**: Computación Peer to Peer	**Sigla**	INF-490
Créditos	3	Conocimientos previos: Sistemas Distribuídos, Redes, Sistemas Operativos.	

Descripción

Este curso es un curso avanzado con énfasis en la propia exploración e investigación. La evaluación se realizará con un certamen y un proyecto. El proyecto está orientado a diseñar e implementar un servicio a gran escala usando técnicas Peer to Peer. El trabajo será en equipo.

Objetivos

+ Presentar al estudiante los algoritmos que existen para manejar sistemas distribuidos de gran escala.
+ Mostrar al alumno la forma de manejar los aspectos relacionados con la tolerancia a falla, disponibilidad del sistema y de seguridad.
+ Presentar aplicaciones existentes en diversos ámbitos tales como: sistemas de bases de datos, sistemas de archivos, correo electrónico, e-business, computación móvil.

Contenidos

1. Problemática de los Sistemas a Gran Escala: historia, problemas de identificación, de partición, latencia, fallas bizantinas.
2. Funciones Hashing: aspectos teóricos y prácticos. Ejemplos MD5, SHA.
3. Técnica Peer to Peer: conceptos, definición y aplicaciones.
4. Capas Peer to Peer: Generalidades (anillo P2P), especificaciones con PASTRY, CHORD, TAPESTRY sus semejanzas y diferencias conceptuales.
5. Ruteo en Peer to Peer: conceptos y ejemplos de ruteo con prefijo, ruteo de saltos exponenciales. Mantención de tablas de ruteo.
6. Tolerancia a Falla: problemas, algoritmos, réplicas. Ejemplo usando Leaf set
7. Seguridad: Manejo de fallas bizantinas, métodos de autentificación de usuarios, encriptación convergente.
8. Conclusiones, investigación actual y aplicaciones

Metodología

El curso es de tipo presencial apoyado por documentación disponible en página WEB . Incluye análisis de artículos recientes y relevantes en el tema.

Evaluación

El certamen tiene una ponderación de un 50% que incluye la revisión de papers. El proyecto tiene una ponderación del 50%.

Bibliografía

[1] P2P Networking Overview: The Emergent P2P Platform of Presence, Identity, and Edge Resources (O'Reilly Research), Kelly Truelove, Clay Shirky, Lucas Gonze, Rael Dornfest, Kelly Truelov, Dale Dougherty, O'Reilly; 1 edition (October, 2001).
[2] Peer-to-Peer: Building Secure, Scalable, and Manageable Networks, Dana Moore, John Hebeler, McGraw-Hill Companies; 1st edition (November 28, 2001).
[3] Peer To Peer Computing: The Evolution Of A Disruptive Technology, Ramesh Subramanian, Brian D. Goodman, Idea Group Publishing (December 31, 2004).

Elaborado	Xavier Bonnaire	**Observaciones**:
Aprobado	Depto. de Informática	
Fecha	01/05/05	

ASIGNATURA: Seguridad de sistemas distribuidos		**SIGLA**: INF-490
Prerrequisitos: Un curso introductorio de seguridad Conocimientos básicos de UML. Los slides estarán en inglés.	**Créditos USM:**	**Créditos SCT:**

OBJETIVOS: La mayoría de los sistemas de información modernos son distribuidos. En este curso se considera los requerimientos de seguridad para este tipo de sistemas, incluyendo sus ataques y sus posibles defensas. Se aplica Unified Moldeling Language (UML) y patrones para describir arquitecturas y mecanismos específicos.

En particular, se abordan sistemas como servicios de red (web services), computación en nube (Cloud computing), y sistemas inalámbricos. Además, se estudia una metodología para construir y evaluar sistemas seguros. Un objetivo importante del curso es proporcionar una perspectiva de cómo la seguridad se coordina con los aspectos funcionales de una aplicación. Otro objetivo es comprender la estructura e implementación de los mecanismos para poder construir y evaluar sistemas seguros.

CONTENIDOS:
1. Motivación y perspectiva.
2. Patrones de seguridad. Patrones de abuso. Arquitecturas de referencia. Revista de UML.
3. Metodologías para aplicaciones seguras distribuidas
4. Arquitecturas distribuidas. Estilos y patrones: Broker, pipes & filters, blackboard, agentes. P2P. Middleware. SOA.
5. SOA y Web services: arquitecturas, ataques, y estandares. Patrones de seguridad para web services. Identidad.
6. Securidad en nubes (cloud computing): amenazas y defensas.
7. Seguridad de sistemas inalámbricos. Estandares de seguridad. Seguridad de redes ad hoc y sensores.
9. Sistemas ciberfísicos. Smart grid (red inteligente). Internet of things.
10 Aplicaciones complejas distribuidas: sistemas médicos, financieros, de transporte

OBJETIVOS:
Los objetivos de aprendizaje son:
1) Conocer y comprender los requerimientos de seguridad que tiene los sistemas de información distribuidos
2) Aplicar UML y patrones para describir aspectos de seguridad en arquitectura de sistemas y de software
3) Conocer una metodología para construir y evaluar sistemas seguros, considerando su estructura y los mecanismos integrados
4) Comprender cómo la seguridad se relaciona con los aspectos funcionales de una aplicación

METODOLOGÍA DE TRABAJO: Clases con tareas de diseño e investigación

SISTEMA DE EVALUACIÓN: Tres tareas (30%) y examen o proyecto final (70%)
Todos se hacen en la casa.

INDICACIONES PARTICULARES: No

BIBLIOGRAFÍA:

E.B.Fernández, E.Gudes, and M. Olivier, The design of secure systems, bajo contrato con Addison-Wesley. (Borrador como notas de clases)

E.B.Fernandez, “Security patterns in practice: Building secure architectures using software patterns”, libro a aparecer en Wiley Series on Software Design Patterns. Mayo 2013. (Borrador como notas de clases)

E.B.Fernández, O.Ajaj, I.Buckley, N.Delessy-Gassant, K.Hashizume, M.M. Larrondo-Petrie, “ A Survey of Patterns for Web Services Security and Reliability Standards”. *Future Internet* 2012, *4*, 430-450. http://www.mdpi.com/1999-5903/4/2/430/

Keiko Hashizume, David G. Rosado, Eduardo Fernández-Medina, Eduardo B. Fernández
“An Analysis of Security issues for Cloud Computing”, accepted for the Journal of Internet Services and Applications, Springer. (SCOPUS)

Keiko Hashizume, Eduardo B. Fernández, and María M. Larrondo-Petrie, “A Reference Architecture for Cloud Computing”, enviada para publicación.

A.V. Uzunov, E.B. Fernández & K. Falkner (2012), "Securing distributed systems using patterns: A survey", Computers & Security, 31(5), 681 - 703. (ISI, IF= 0.868))
doi:10.1016/j.cose.2012.04.005

A.V. Uzunov, E.B. Fernandez & K. Falkner , “Engineering Security into Distributed Systems: A Survey of Methodologies”, accepted for the Journal of Universal Computer Science (ISI)

[Sta12] W. Stallings and L. Brown, Computer security: Principles and practice (2nd Ed.), Pearson 2012.

ELABORADO **APROBADO** **FECHA**	Dr. Eduardo Fernández 26-12-2012	**OBSERVACIONES**:

ACTUALIZADO **APROBADO** **FECHA**	VERSIÓN ACTUALIZADA	**OBSERVACIONES**:

Asignatura	Tema Especial en Informática: Astro-Informática	**Sigla**	INF-490
Créditos SCT	7	**Conocimientos previos:** ILI-236, FIS-120	

Descripción

Corresponde a un curso electivo del área de Sistemas Computación, donde los alumnos aprenderán temas fundamentales para comprender astronomía, y cuáles son sus implicancias dentro de la astro-ingeniería, y en particular en la Ingeniería Informática. El objetivo general de este curso es dar al alumno suficiente conocimiento para saber cuáles son los diversos tipos de telescopios, la tecnología desarrollada para realizar observaciones astronómicas, los sistemas computacionales envueltos, y las características principales y más relevantes de los distintos proyectos de astronomía desde una perspectiva de la Ingeniería Informática.

Objetivos

- Explicar movimientos terrestres y celestes.
- Realizar observaciones en telescopios virtuales (ópticos y radio).
- Conocer estándares de Observatorios Virtuales (OV).
- Explicar distintos tipos de observaciones astronómicas (óptica, radio, alta energía) y su relación con la informática.
- Crear implementaciones básicas de observatorios virtuales (IVOA).
- Conocer el estado actual de la astro-ingeniería, y cuáles son los actuales y próximos desafíos en el área (data mining sobre OV)

Contenidos

1. Historia y fundamentos teóricos de la astronomía y sus instrumentos.
2. Observatorios Virtuales (OV) y diferentes tipos de Telescopios.
3. Algoritmos para búsqueda en OV
4. Algoritmos para procesamiento sobre OV (CASA)
5. Estado de la Práctica
6. Nuevos desafíos

Metodología

1. Clases expositivas con apoyo de medios audiovisuales.
2. Utilización de telescopios virtuales para realizar observaciones con datos de telescopios almacenados.
3. Visitas a observatorios de mediana envergadura para realizar observaciones y conocer el software que utilizan en él.
4. Familiarización y exposición de un tema de actualidad en astro-ingeniería y su relación con la ingeniería informática.
5. Taller de desarrollo de aplicaciones en observatorios virtuales

Evaluación

C1 = Nota Certamen 1; C2 = Nota Certamen 2; TA = Nota Talleres; E = Nota Exposición; P = Nota Participación.
Calificación Nota Final C1*0.25 + C2*0.25 + TA*0.30 + E*0.10 + P*0.10

Bibliografía

[1] ACS Workshop 2007, 2008, 2009, 2010, UTFSM, ESO, NRAO, NAOJ, JAO. http://acsworkshop.inf.utfsm.cl/

[2] ALMA Common Software 8.0 -- Documentation, ALMA Project http://www.eso.org/projects/alma/develop/acs/OnlineDocs/index.html
[3] ACS C++ Component/Container Framework Tutorial, by Grega Milcinski, Matej Sekoranja, Bernhard Lopez, David Fugate, Alessandro Caproni.
[4] ALMA Common Software and Python, by David Fugate.
[5] ACS Java Component Programming Tutorial, by Heiko Sommer.
[6] Astronomical Algorithms, Second Edition, Jean Meeus, Willmann-Bell Inc. Publishers, 2005.
[7] A. Richard Thompson, J.M. Moran, G. W. Swenson, Jr. Interferometry and Synthesis in Radio Astronomy, second edition (2001).
[8] Telescope Control, Second Edition, Mark Trueblood and Russel Merle Genet, 1997.
[9] Telescope Control Systems, Krewalk et al., United State Patent 4682091, 1987.
[10] The SAO/NASA Astrophysics Data System, http://www.adsabs.harvard.edu/
[11] VLT Common Software 2009 -- Documentation Kit, European Southern Observatory. http://www.eso.org/projects/vlt/sw-dev/wwwdoc/VLT2009/dockit.html

Elaborado:	Mauricio Solar	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	24-08-2012	

Asignatura	Temas Especiales en Informática: Ingeniería de Software Experimental	**Sigla**	INF-492
Créditos	3	**Conocimientos previos**: Ingeniería de Software	

Descripción

Experimentación en Ingeniería de Software concierne el uso del diseño y análisis experimental para validar ideas y creencias, en un campo ampliamente dominado por suposiciones y especulaciones, orientado en forma práctica para los ingenieros de software. ¿Son válidas nuestras suposiciones? ¿Qué afirmaciones de la comunidad de desarrollo de software son válidas? ¿Bajo qué circunstancias son válidas? Responder estas preguntas es crítico para otorgar mayor certeza a las ideas en que se fundamenta la Ingeniería de Software. Durante la construcción de software no se utilizan, habitualmente, técnicas formales de experimentación. Este hecho contrasta con las prácticas comunes de otras ingenierías y campos científicos, en las cuales es obligatorio realizar una rigurosa experimentación que apoye las investigaciones realizadas. En este curso se discute el uso de Análisis y Diseño de Experimentos en Ingeniería de Software, estableciendo las bases teóricas para la efectiva realización de experimentos.

Objetivos

Al finalizar el curso el alumno:

1. Comprenderá la importancia de validar empíricamente las técnicas usadas en el desarrollo de software
2. Sabrá aplicar diversas técnicas de experimentación en Ingeniería de Software
3. Conocerá algunas experiencias reales de experimentación en Ingeniería de Software
4. Conocerá líneas de investigación actuales y futuras en Ingeniería de Software Experimental

Contenidos

1. Introducción a la experimentación en Ingeniería de Software
2. Principales tipos de diseños y análisis de experimentos
3. Estudios experimentales reales realizados en Ingeniería de Software (diseño, ejecución, análisis crítico)
4. Estado del arte en Ingeniería de Software Experimental (líneas de investigación actuales y futuras, desafíos de la disciplina)

Metodología

- Clases tipo seminario de 90 minutos, combinando exposición con discusión grupal
- Lectura sistemática guiada de artículos fundamentales y capítulos de libros, elaboración de comentarios semanales
- Selección de un tópico de profundización, selección de bibliografía relevante, elaboración de informes periódicos quincenales e informe final
- Diseño, ejecución y análisis crítico de estudios experimentales en Ingeniería de Software

Evaluación

- Informes periódicos/comentarios semanales/quincenales: 30%
- Informe final: 30%
- Presentación oral: 30%
- Participación: 10%

Bibliografía

[1] Basics of Software Engineering Experimentation. Natalia Juristo, Ana María Moreno. Kluwer, 2001

[2] Experimentation in Software Engineering: An Introduction. Claes Wohlin, Per Runeson, Martin Höst. Springer, 1999
[3] Empirical Software Engineering Issues. Vic Basili et al (editors). Lecture Notes in Computer Science 4336. Springer-Verlag, 2007

Conferencias

[1] Empirical Software Engineering and Measurement (ESEM)

Revistas

[1] Journal of Empirical Software Engineering (ESE)
[2] IEEE Software

Elaborado:	M. Visconti	**Observaciones**:
Aprobado:	20-07-2007	
Fecha:	17/07/2007	

PROGRAMA DE ASIGNATURAS NIVEL 500

ASIGNATURA: **Tema Especial en Informática: Métodos Numéricos Computacionales**		**SIGLA**: INF-510
Prerrequisitos: Lenguajes de Programación, Cálculo, Algebra Lineal Numérica, Computación Científica	Créditos SCT: 8	
Horas Semanales Cátedra: 4	Horas Semanales Ayudantía:	Horas Semanales Lab : 2

OBJETIVOS: Discutir el uso acoplado de herramientas clásicas y modernas en la computación científica. Dejando al estudiante capacitado para:

- Evaluar de diferentes algoritmos avanzados para la resolución de sistemas de ecuaciones lineales, comprender ventajas y desventajas de cada uno de los algoritmos propuestos y entender técnicas especiales recientes para mejorar su desempeño.
- Discutir, analizar y proponer aproximaciones a problemas descritos por ecuaciones diferenciales.
- Proponer mejoras, discutir y evaluar de métodos rápidos en Computación Científica.

CONTENIDOS:

1. Visión general del área
2. Algebra Lineal Computacional
   a. Matrices: Completas, Ralas y Libres.
   b. Gradiente Conjugado
   c. GMRes: Reiniciación, Precondicionadores y Convergencia
   d. Métodos de Precondicionadores
3. Introducción a Métodos de Perturbación
   a. Perturbación Regular
   b. Perturbación Singular
   c. Análisis de Capa de Frontera
4. Valores Propios, Funciones Propias y Ecuaciones Integrales.
   a. Expansiones Ortogonales
   b. Ecuaciones Integrales
   c. Funciones de Green
5. Diferencias Finitas y Métodos Espectrales
   a. Introducción
   b. Forma Matricial de Diferencias Finitas
   c. Mallas Periódicas: DFT y la FFT
   d. Suavidad y Precisión Espectral
   e. Matrices de Diferenciación de Chebyshev
   f. Problemas de Valor de Frontera
   g. Problemas de Valor Inicial y Método de las Líneas
6. Tópicos Especiales (Si el tiempo lo permite)
   a. Introducción a Métodos rápidos
   b. Transformada Rápida de Fourier
   c. Método Rápido de Multipolos en 2D

METODOLOGÍA DE TRABAJO:
Clases expositivas con participación del profesor y de los estudiantes. Discusión en grupos.

SISTEMA DE EVALUACIÓN:
30% Tareas, 40% Exámenes y 30% Monografía (Trabajo Final)

Nota: Consideración especial, dependiendo de la cantidad de estudiantes, el examen será take-home con exposición individual.

BIBLIOGRAFÍA:

1. Numerical Analysis, Second Edition, Timothy Sauer, Pearson, 2011, ISBN: 0321783670.
2. Meshfree Approximations Methods with MatLab, Gregory E. Fasshauer, World Scientific Publishing Company, 2007, ISBN: 9812706348.
3. Applied Mathematics, J. David Logan, Third Edition, Wiley, 2006, ISBN: 0471746622.
4. Iterative Methods for Sparse Linear Systems, Yousef Saad, Second Edition, SIAM, 2003, ISBN: 9780898715347.
5. Spectral Methods in MatLab, Lloyd N. Trefethen, SIAM, 2000, ISBN: 0898714656.
6. Applied Numerical Linear Algebra, James W. Demmel, SIAM, 1997, ISBN: 0898713897.
7. Numerical Linear Algebra, Lloyd N. Trefethen and David Bau, III, SIAM, 1997, ISBN: 0898713617
8. Domain Decompositions Methods, Barry Smith, Petter Bjorstad and William Gropp, Cambridge University Press, 1996, ISBN: 0521602866.

ELABORADO **APROBADO** **FECHA**	Claudio Torres López	**OBSERVACIONES:**

ACTUALIZADO **APROBADO** **FECHA**	27 de junio de 2013 12 agosto 2013	**OBSERVACIONES:**

Asignatura	Compresión de Texto	**Sigla**	INF-520
Créditos SCT	8	**Conocimientos previos:** Estructuras de datos	
Hrs. Cat. Sem. : 3	Hrs. Ayud. Sem. : 0	Hrs. Lab. Sem. : 0	

Descripción
Al finalizar este curso el alumno conocerá los fundamentos de teoría de la información, será capaz de reconocer qué tipos de textos pueden ser comprimidos, conocerá las distintas familias de algoritmos de compresión, conocerá a fondo las principales herramientas de compresión de texto disponibles en la actualidad, y será capaz de modificar las herramientas de compresión actuales para agregarles mejoras o proponer nuevas variantes.

Objetivos
1. Entregar los fundamentos de teoría de la información relacionados a compresión de secuencias de texto sin pérdida.
2. Estudiar las familias principales de compresores de texto.
3. Estudiar las principales herramientas de compresión de texto del estado del arte.

Contenidos
1. **Teoría de la Información**: Complejidad de Kolmogorov y sus limitaciones. Entropía, entropía relativa e información mutua. Teorema de Shannon. Desigualdad de Jensen y otras de teoría de la información. Propiedad de equipartición asintótica, entropía empírica. Procesos estocásticos, cadenas de Markov y su entropía. Procesos ergódicos. Compresión universal, convergencia y redundancia.

2. **Compresión Estadística**: Modelamiento y codificación. Modelamiento estático, semiestático y dinámico. Códigos y sus propiedades. Desigualdad de Kraft. Códigos óptimos y redundancia. Códigos de Fano y de Shannon. Códigos de Huffman. Optimalidad. Codificación artimética. Códigos dinámicos o adaptativos: Huffman y aritméticos. Modelamiento de lenguaje natural. Códigos multiarios. Códigos densos y su uso para bases de datos de texto. Compresión PPM.

3. **Compresión Basada en Diccionarios**: Compresión por sustitución de substrings. Compresión Lempel-Ziv: LZ77, LZ78, LZW y variantes. Universalidad y convergencia. Compresión basada en gramáticas. Re-Pair, Sequitur. Compresión de texto estructurado.

4. **Compresión basada en Transformación**: Transformación de Burrows-Wheeler. Move-to-front y run-length encoding. Compresión de árboles rotulados.

5. **Herramientas de Compresión**: gzip, bzip, lzma, ppm-d, google snappy. Comparación práctica entre ellas.

Metodología
El curso se desarrollará mayoritariamente mediante clases magistrales.

Evaluación
Individual, mediante dos certámenes y la entrega de un informe de investigación y una tarea (implementación + evaluación de un método de compresión, o modificación de una herramienta de compresión existente).

Bibliografía

[1] T. Cover and J. Thomas. Elements of Information Theory. Wiley, 2nd ed, 2006.
[2] T. Bell, J. Cleary, and I. Witten. Text Compression. Prentice-Hall, 1990.
[3] I. Witten, A. Moffat, and T. Bell. Managing Gigabytes. Morgan Kauffmann, 2nd ed, 1999.
[4] D. Salomon. Data Compression: The Complete Reference. Springer, 4th ed, 2007.

Revistas

1. Jeffrey Scott Vitter: Design and analysis of dynamic Huffman codes. J. ACM 34(4): 825-845 (1987)
2. Ian H. Witten, Radford M. Neal, John G. Cleary: Arithmetic Coding for Data Compression. Commun. ACM 30(6): 520-540 (1987)
3. Jacob Ziv, Abraham Lempel: A Universal Algorithm for Sequential Data Compression. IEEE Transactions on Information Theory 23(3): 337-343 (1977)
4. Jacob Ziv, Abraham Lempel: Compression of Individual Sequences via Variable-Rate Coding. IEEE Transactions on Information Theory 24(5): 530-536 (1978)
5. Burrows M and Wheeler D (1994), *A block sorting lossless data compression algorithm*, Technical Report 124, Digital Equipment Corporation
6. Paolo Ferragina, Fabrizio Luccio, Giovanni Manzini, S. Muthukrishnan: Compressing and indexing labeled trees, with applications. J. ACM 57(1): (2009)

Elaborado por:	Diego Arroyuelo	**Observaciones:**
Fecha Emisión:	11/01/2012	
Fecha Aprobación por Comité I&P:	16-01-2012	

Asignatura	Evaluación de Arquitectura de Software	**Sigla**	INF-524
Créditos SCT	8	**Conocimientos previos:**	

Descripción

Exploración sistemática de literatura científica reciente relativa a la medición, comparación y evaluación de arquitecturas de software.

Objetivos

Al aprobar el curso el alumno podrá:

evaluar, comparar y mejorar especificaciones de arquitectura y las estructuras que ellas denoten, con criterios de calidad intrínseca (técnicos) y extrínsecos (objetivos)

elaborar y describir modelos de calidad para arquitecturas de software

distinguir los problemas de arquitectura de software de los que no lo son, y explicar articuladamente las nociones básicas de la disciplina

proponer formas de evaluación de arquitecturas para situaciones concretas

Contenidos

1. Modelos de calidad para arquitecturas de software
2. Evaluación de arquitecturas de software
3. Comparación de arquitecturas de software

Metodología

- Lectura sistemática de artículos de la literatura científica, reciente y/o fundamental.
- Talleres de técnicas específicas (e.g. árboles de utilidad).

Evaluación

- Evaluación de los talleres de técnicas específicas.
- Artículo publicable en conferencia internacional (indexada o no).

Bibliografía

[1] *Evaluating Software Architectures: Methods and Case Studies*. P.Clements, R.Kazman, M.Klein. Addison-Wesley Professional (2001). ISBN-10: 020170482X.

[2] "A Basis for Analyzing Software Architecture Analysis Methods". Kazman, Bass, Klein, Lattanze, Northrop. *Software Quality Journal*, 13, 329–355, 2005

[3] *ISO/IEC 25010, Software product Quality – Requirements and Evaluation (SQuaRE) — Quality Model*. ISO – International Standards organization, Geneva (2006).

[4] "Five Ontological Levels to Describe and Evaluate Software Architectures". H.Astudillo. Revista de la Facultad de Ingeniería de la Universidad de Tarapacá (2004).

[5] "Standard quality model to design software architecture". F.Losavio. Journal of Object Technology, 1(4), 2002, pp.165-178.

[6] "Tools and Methods for Evaluating the Architecture". Software Engineering Institute, Carnegie Mellon University. http://www.sei.cmu.edu/architecture/tools/evaluate/. Last visited: 2012-01-02

Revistas

[1] Proceedings de WICSA, QoSA, ECSA, FSE, ICSE

[2] Lecture Notes in Computer Science (LNCS), Springer

Elaborado:	20 Julio 2006	**Observaciones:**
Aprobado:	05-09-2007	
Fecha:		**Actualizado Enero de 2013.**

Asignatura	Programación Paralela Avanzada		**Sigla**	INF-534
Créditos SCT	4	**Conocimientos previos:** Programación en C, Sistemas Operativos, Redes.		

Descripción

Este curso es un curso dedicado a la programación paralela con las últimas arquitecturas paralelas existentes (procesadores multi-core, clusters, grids). El objetivo del curso es ser capaz de programar un algoritmo en paralelo en el marco de una tesis de Magister o de Doctorado, así que escribir un short paper para presentar los resultados obtenidos.

Objetivos

Presentar las arquitecturas modernas para la programación paralela.
Presentar los problemas de paralelización y los métodos para resolverlos.
Saber detectar y solucionar los problemas de sincronización y de secciones críticas.
Saber programar un algoritmo usando los Threads POSIX con un procesador multi-core..
Saber programar con el interfaz de programación paralela MPI (Message Passing para redes locales y clusters).
Saber programar usando OpenMP para una paralelización semi-automática del código.
Programación de un algoritmo paralelo (a elección) y presentación de los resultados en un short paper.

Contenidos

1. Introducción
   - 1.1. Los computadores paralelos obsoletos.
   - 1.2. Las arquitecturas modernas para la programación paralela (Multicore, Clusters, Grids).
   - 1.3. Objetivos de la programación paralela
2. Paralelización
   - 2.1. Noción de Speedup, Leyes de Amdahl y Gustafson.
   - 2.2. Técnicas de paralelización (decomposición).
   - 2.3. Paralelización automática.
   - 2.4. Pensar Paralelo (mejorar el grado de paralelismo).
   - 2.5. Modelos de Memoria Compartida, Message Passing.
   - 2.6. Lo que se puede paralelizar
   - 2.7. Lo que no se puede paralelizar.
3. Gestión de recursos compartidos
   - 3.1. Accesos concurrentes (Inconsistencia de los datos).
     - 3.1.1.Ejemplos
   - 3.2. Secciones críticas y exclusión mutua
     - 3.2.1.Semáforos y secuencialización
4. Sincronización.
   - 4.1. ¿Por qué sincronizar?
   - 4.2. Barriers, Rendez-Vous
   - 4.3. Uso de los semáforos
   - 4.4. Modelo Productor / Consumidor (1-1, 1-C, P-C)
5. Programación Multi-Processos con LINUX / UNIX
   - 5.1. Funcionamiento de la Memoria Compartida (Interfaz).
   - 5.2. Interfaz de programación con semáforos en LINUX / UNIX
   - 5.3. Ejemplo del pasaje de avión
   - 5.4. Ejemplo de un cálculo matricial
6. Programación Multi-Threads con LINUX / UNIX
   - 6.1. Definición de un Thread (diferencias con el multi-processos).
   - 6.2. Los Threads POSIX (biblioteca de programación).

6.3. Modelos de programación con Threads (Pool de Threads, Master/Slave)
7. Programación con Message Passing (Redes, Clusters, …).
   7.1. Modelo del Message Passing.
   7.2. Diferencia con la memoria compartida.
   7.3. Biblioteca de programación. MPI (Message Passing Interface).
8. Programación con OpenMP
   8.1. Conceptos de OpenMP (Directivas de compilación).
   8.2. Paralelización Semi-Automática
9. Herramientas de programación
   9.1. IDE.
   9.2. Debuggers (ddd, Valgrin, Dbmalloc).
   9.3. Profilers (Gcc, Gprof).

Metodología

La metodología usada para el curso será la siguiente:

- Clase magistral con el apoyo de apuntes y de un espacio Web con foros de conversación
- Programación de un algoritmo en paralelo (algoritmo a elección).
- Escritura de un Short-Paper para presentar los resultados obtenidos
- Presentación del artículo durante un seminario de investigación

Evaluación

La nota final del curso de calculará de la siguiente forma:

- Dos certamenes: 50%
- Redacción de un Short Paper: 40%
- Presentación del Short-Paper: 10%.

Bibliografía

[1] Grama A., Gupta A., Karypis G., Kumar V., Introduction to Parallel Computing, Second Edition, Addison Wesley, 2003.
[2] Herlihy M., Shavir N., The Art of Multiprocessors Programming, Morgan Kaufmann, 2008.
[3] Stevens W.R., Rago S. A., Advanced Programming in the UNIX Environment, Second Edition, Addison Wesley, 2008.

Elaborado:	Xavier Bonnaire	**Observaciones:**
Aprobado:	Comité IP del DI	
Fecha:	22-04-2009	

Asignatura	Modelos computacionales en Series de Tiempo		**Sigla**	INF-560
Créditos SCT	8	**Conocimientos previos:** Lenguajes de programación, Computación Científica, Estadística Computacional e Introducción a las máquinas de aprendizaje.		

Descripción

Este curso pretende introducir a los alumnos en el análisis de series de tiempo y sus aplicaciones. Se estudiarán los fundamentos teóricos del problema de pronóstico de series de tiempo, tanto desde el punto de vista estadístico, como computacional; además, se abordaran los fundamentos probabilísticos de los modelos clásicos de series de tiempo y aquellos basados en máquinas de aprendizaje. Serán tratadas las metodologías más relevantes para construir modelos de series de tiempo poniendo énfasis en aspectos computacionales y se abordarán diversas metodologías de validación.

Objetivos

Al término del curso el alumno debe ser capaz de:

- Comprender los fundamentos teóricos del análisis de series de tiempo.
- Conocer las metodologías clásicas de análisis de series de tiempo y sus limitaciones.
- Aplicar los modelos clásicos a problemas de series no estacionarias y de alta frecuencia.
- Aplicar las metodologías de análisis de series de tiempo basados en máquinas de aprendizaje en diversos problemas reales.
- Diseñar modelos de series de tiempo y validarlos empíricamente.

Contenidos

1. Características de una serie de tiempo.
- Fundamentos teóricos.
- Procesos estocásticos.
- Teorema de descomposición de Wold.
- Modelos estadísticos de series de tiempo.
- Series de tiempo estacionarias y no estacionarias.

2. Inteligencia computacional
- Introducción.
- Computación flexible.
- Algoritmos probabilísticos.
- Computación Evolutiva.
- Métodos Híbridos.
- Lógica Fuzzy.
- Aplicaciones.

3. Modelos clásicos.
- Introducción.
- Análisis exploratorio de datos.
- Transformaciones en series de tiempo.
- Modelos ARMA y ARIMA.
- Ecuaciones en diferencias.
- Modelos integrados para datos no estacionarios.
- Identificación, Estimación y Pronóstico.

- Construcción de modelos.
- Validación de modelos.
- Otros modelos ARFIMA, GARCH (opcional).
- Aplicaciones.

4. Máquinas de aprendizaje en Series de Tiempo

- Introducción.
- Redes Neuronales de multicapa.
- Máquinas de soporte vectorial.
- Modelos NARMA.
- Pronóstico de Series Multivariadas.
- Series de Tiempo Caóticas.

5. Temas opcionales.

- Modelos de espacios de estados.
- Modelos estadísticos en el dominio de la frecuencia.
- Series de tiempo de alta frecuencia.
- Tópicos de finanzas computacionales.
- Series de tiempo multidimensionales.
- Pronóstico de series de tiempo mediante Lógica Fuzzy.
- Modelos Neuro-Fuzzy en series de tiempo.
- Predicción de series georeferenciadas (Kriging).
- Análisis de correlación de procesos espaciales.

Metodología

El curso se desarrollará con clases expositivas de 90 minutos. El alumno deberá realizar tareas individuales para complementar su aprendizaje.
Durante el desarrollo del curso el alumno deberá seleccionar lecturas relevantes de artículos y elaborar un proyecto de investigación en algún tópico de su elección en forma de seminario y discusión del estado del arte. Este proyecto será evaluado mediante una presentación oral y un reporte escrito.
El examen final del curso estará basado en los tópicos tratados en clases por los profesores.

Evaluación

Tareas	30%
Proyecto	40%
Examen	30%

Bibliografía

[1]C. Bishop. Pattern Recognition and Machine Learning, Springer Verlag 2006.
[2]R. Shumway and D. Stoffer. Time Series Analysis and Its Applications with R Examples, 3rd edition, Springer 2011.
[3]A. Palit and D. Popovic. Computational Intelligence in Time Series Forecasting: Theory and Engineering Applications, 1st edition. Springer Verlag 2005.
[4]G. Box, G. Jenkins and G. Reinsel. Time Series Analysis: Forecasting and Control, 4th edition, Wiley, 2008.
[5]R. Tsay. Analysis of Financial Time Series, Wiley, 3rd edition, 2010.

Elaborado:	Héctor Allende	**Observaciones:** **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	02/07/2010	

Asignatura	Métodos Computacionales en Teoría de Funciones	**Sigla**	INF-562
Créditos SCT	4	**Conocimientos previos:** - Contenidos asignaturas ILI-285 e ILI-286: Computación Científica I y II. - Autorización del Profesor.	

Descripción

Durante la última década, la ciencia de la computación y la teoría de la aproximación han llegado a integrarse en un área de la investigación, que subsume aspectos relevantes de varias disciplinas, tales como las redes neuronales y sus derivados, el diseño asistido por computador, el procesamiento de imágenes, las funciones especiales de la Física-Matemática, las transformaciones conformes, las "wavelets", los fractales, y otros sectores de la ciencia, la técnica y la matemática aplicada. El objetivo de este curso es discutir, y aplicar a situaciones concretas, los resultados más recientes de esta activa área de la investigación en la frontera entre la ciencia de la computación y la teoría de la aproximación.

Competencias (TRANSVERSALES y/o ESPECÍFICAS) a las que contribuye:

1. Capacidad para identificar, plantear y resolver problemas en el contexto de los métodos computacionales y de la teoría de la aproximación
2. Capacidad de aplicar los conocimientos en la práctica.
3. Capacidad de abstracción y modelamiento.

Objetivos

Aplicar métodos y estrategias de diseño de algoritmos a problemas en el ámbito de la aproximación real y compleja.

Aplicar esos métodos y algoritmos a problemas de la ciencia y la técnica.

Determinar la complejidad promedio y los peores casos de esos algoritmos.

Contenidos

1. Elementos de la teoría de funciones y de la aproximación.
2. Métodos computacionales en la teoría de funciones y de la aproximación
3. Métodos computacionales en la teoría de la convolución.
4. Problemas computacionales asociados a la función zeta de Riemann.
5. Análisis de los algoritmos estudiados. Complejidad.

Metodología

Periódicamente habrá clases expositivas, desarrollo de ejemplos, y tareas (tanto teóricas como de programación).
Discusiones técnicas y preparación de artículos y monografías.
Exposiciones de problemas específicos por parte de los participantes.
NOTA: La asignatura se dictará en la modalidad de tutoría durante el segundo semestre 2010.
Dedicación a la asignatura

Exposición del profesor	20 Hrs.
Estudio Personal	50 Hrs.
Confección de Informes	20 Hrs.
Resolución de Problemas	60 Hrs.
Total	150 Hrs.

Evaluación

Disertaciones individuales de los alumnos acerca de temas escogidos de los contenidos temáticos.

Redacción de artículos y monografías
Presentación y defensa de los resultados resumidos

Calificación
Promedio de las notas obtenidas en los instrumentos que se apliquen.

Bibliografía
[1] St. Ruscheweyh et al., Computational Methods and Function Theory World Scientific, Singapur, 2009/2005/2001/1997/1994.
[2] St. Ruscheweyh, L. Salinas. "New Pólya-Schoenberg type theorems". Journal of Mathematical Analysis and Applications, 363 (2) (2010), pp. 481-496.
[3] St. Ruscheweyh, L. Salinas. "Universally Prestarlike Functions as Convolution Multipliers". Mathematische Zeitschrift, 263 (3) (2009) , pp. 607-617.
[4] St. Ruscheweyh, L. Salinas, T. Sugawa. "Completely Monotone Sequences and Universally Prestarlike Functions". Israel Journal of Mathematics, Volume 171, Number 1 (2009), pp. 285-304.
[5] R. Fournier, L. Salinas. “On a question of Brezis and Korevaar concerning a class of square summable sequences”. In Banach Spaces of Analytic Functions, Rita A. Hibschweiler and Thomas H. MacGregor, Eds. American Mathematical Society, Contemporary Mathematics, Vol. 454, 35-42, April 2008.

Elaborado:	Luis Salinas	**Observaciones:**
Aprobado:	Depto. Informática	
Fecha:	03-08-2010	

Asignatura	Diseño Avanzado de Algoritmos	**Sigla**	INF-564
Créditos SCT	8	**Conocimientos previos:** - Estructuras de Datos	

Descripción
El curso está orientado a desarrollar los conocimientos y habilidades asociadas al diseño avanzado de algoritmos. Es una asignatura auto contenida en el área, por lo que no considera pre-requisitos. En ella se tratan las técnicas básicas de diseño de algoritmos y además, se profundiza en tres aspectos avanzados de diseño: algoritmos aproximados, aleatorizados y en línea.

Objetivos
1. Analizar y diseñar algoritmos que permitan resolver problemas computacionales.
2. Evaluar la pertinencia de algoritmos en la resolución de problemas computacionales.
3. Potenciar la capacidad analítica en la resolución de problemas en la informática.

Contenidos
1. Técnicas de Análisis de Algoritmos
   - Notación asintótica, análisis caso promedio, análisis amortizado, funciones generatrices.
2. Técnicas Básicas de Diseño de Algoritmos
   - Divide y vencerás: MergeSort, conteo de inversiones, producto de enteros, convex hull, convolución y FFT.
   - Decrece y vencerás: Búsqueda con poca información (galloping search), búsqueda interpolada, búsqueda con recursos acotados. QuickSelect.
   - Programación dinámica: mínimos cuadrados segmentado, subsets sum, mochila y extensión a RNA, alineación de secuencias, producto matricial.
3. Algoritmos en línea
   - Análisis de competitividad.
   - Problemas: predicción de demanda (productos, páginas en OS), predicción de valores.
4. Algoritmos Aproximados
   - P/NP: reducciones de tiempo polinomial, SAT, coloreo de grafos, vertex cover.
   - Diseño con cotas de aproximación: balance de carga, set cover, center selection, disjoint paths.
   - Búsqueda local: metrópolis, simulated annealing, dinámica y equilibrio de Nash.
5. Algoritmos Aleatorizados
   - Tipología: Montecarlo, Las Vegas, ejemplos.
   - Cotas de Chernoff, balance de carga, ruteo, caching.
   - Paralelismo: modelo PRAM, ejemplos. Metodología

Metodología
La asignatura se desarrollará con clases expositivas de 90 minutos. Se exponen los conceptos y se desarrollan problemas en clases.

Evaluación
1. 2 certámenes 40% cada uno
2. 2 tareas 20% en total
- Nota mínima en promedio de tareas: 55

Bibliografía
Básica

1. Levitin, A. (2011). Introduction to the design and analysis of algorithms, 3rd Ed., Pearson.
2. Kleinberg, J., Tardos, É. (2013). Algorithms design, 1st Ed., Pearson.

Recomendada

1. Cormen, Th., Leiserson, Ch., Rivest, R., Stein, C. (2009). Introduction to algorithms, 3rd Ed., MIT Press.

Elaborado:	Diego Arroyuelo y Marcelo Mendoza	**Observaciones:** - Asignatura Obligatoria para el Programa Magíster en Ciencias de la Ingeniería Informática - Asignatura Obligatoria para el Programa Doctorado en Ingeniería Informática
Aprobado:	Depto. Informática	
Fecha:		

Asignatura	Metodología de la Investigación.		**Sigla**	INF-565
Créditos SCT	8	**Conocimientos previos:**		

Descripción

Este curso introduce a los alumnos en la metodología de la investigación científica en Ingeniería Informática y les permite desarrollar algunas competencias necesarias en la formulación de proyectos de investigación y en la escritura de artículos y tesis.

Objetivos

Al finalizar este curso, los alumnos:

- Conocerán los diferentes tipos de investigación teórica, aplicada y experimental.
- Conocerán las principales visiones de la filosofía de la ciencia.
- Serán capaces de diseñar experimentos computacionales.
- Serán capaces de preparar artículos de investigación, reportes técnicos y tesis.
- Serán capaces de formular proyectos de investigación.

Contenidos

1. Epistemología de la ciencia.

- Introducción: tipos de ciencias (natural, artificial, exactas, etc.).
- Sociología de la ciencia: disciplinas y comunidades, praxis científica, rol de la publicación.
- Filosofía de la ciencia.
    i. Empirismo.
    ii. Positivismo y demarcación.
    iii. Popper y falsasionismo.
    iv. Kuhn y paradigmas.
    v. Lakatos y programas de investigación.
    vi. Feyerabend y la argumentación contra el método.
    vii. Perspectiva bayesiana de la ciencia.
- Inferencia, inducción, abducción.

2. La investigación científica.

- La investigación: definición, características, formas y tipos.
- Método científico tradicional: elementos, etapas, características.
- La ética en la investigación científica.
- Investigación empírica en ciencias de la computación.
    i. El proceso de la investigación empírica.
    ii. Tipos de investigación empírica: métricas, surveys, estudios de casos, cuasi-experimentos, experimentos y simulación.
- Investigación científica como empresa cooperativa.
    i. Gestión de proyectos de investigación.
- Formulación de proyectos: formulación, preguntas de la investigación y justificación de la investigación.
    i. Definición del problema, hipótesis de trabajo, objetivos, metodología y plan de trabajo, divulgación de los resultados.
    ii. Elaboración del marco teórico: revisión de la literatura, tipos de fuentes bibliográficas y recursos para la búsqueda.

iii. Experimentación: unidad experimental, factor, tratamientos, bloqueos, variables dependientes, variables independientes, variables de bloqueo, sujetos, hipótesis, validación y replicación.

3. Experimentos computacionales.

- Principios del diseño experimental.
- Experimentos en informática: diseño, ejecución, interpretación, reporte.
- Análisis de datos.
    i. Análisis estadístico de datos.
    ii. Análisis descriptivo.
    iii. Hipótesis estadística.
    iv. Análisis paramétrico y no paramétrico.
    v. Métodos de reconocimiento de patrones.
- Relación entre hipótesis y evidencia: el problema de la validación de la hipótesis.
- Diseño de experimentos en subdisciplinas de la Informática
    i. Algoritmos Probabilísticos.
    ii. Ingeniería de Software.
- Investigación reproducible

4. Elaboración de informes científicos.

- Resultados intermedios del proceso científico: reporte técnico, artículo (workshops, conferencia, revista), tesis.
- Desarrollo de la argumentación.
- Calidad de un documento: claridad, precisión, fluidez, objetividad, comunicación eficiente.

Metodología:
El Curso se desarrollará con clases expositivas de 90 minutos. Durante el desarrollo del curso se entregará lecturas relevantes de artículos para complementar el aprendizaje de los alumnos. Éstos también deberán elaborar un proyecto de investigación y un artículo en algún tópico de su interés. El examen final del curso será basado en una presentación oral y un reporte escrito de su tema de investigación.

Evaluación:

- Lectura y análisis de artículos de investigación. 25%
- Formulación de un proyecto de investigación. 35%
- Escritura de un artículo científico. 40%

Bibliografía:

[1] T. Bartz-Beielstein “Experimental Research in Evolutionary Computation” Ed. Springer 2010.
[2] S. Bem, H. L. de Jong: *Theoretical Issues in Psychology: An Introduction (2nd ed.)*. Sage Publications (2006).
[3] Kai-Tai Fang, Runze Li and A. Sudjianto, T. “Design and Modeling for Computer Experiment ” Ed. Chapman Hall/CRC Springer 2006.
[4] N. Juristo, A. M. Moreno: *Basics of Software Engineering Experimentation*. Kluwe Academic Publishers (2001).
[5] E. M. Phillips, D. S. Pugh: *How to get a PhD, a handbook for students and their supervisors*. Open University Press (2010).
[6] A. F. Chalmers, “ What is this thing called Science” University of Queensland Press, revised edition (1999).

[7] P. P. Balestrassi, E. Popova, A. P. Paiva, J. W. Marangon Lima: *Design of experiments on neural network's training for nonlinear time series forecasting*.
[8] K. Hinkelmann, O. Kempthorne: *Design and Analysis of Experiments, Volume I: Introduction to Experimental Design.* John Wiley & Sons (1994).

Elaborado:	Héctor Allende; Hernán Astudillo; Alejandro Frery	**Observaciones:** **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	11/06/2010	

Asignatura	Simulación mediante Mallas Geométricas	**Sigla**	INF-568
Créditos SCT	8	**Conocimientos previos:** MAT-023	

Descripción

Esta asignatura forma parte de los cursos electivos de los programas de posgrado científico del Departamento de Informática y forma parte del área de conocimientos Ingeniería Aplicada. Esta asignatura está orientada a alumnos de Magíster y Doctorado en Ciencias de la Ingeniería Informática. En esta asignatura los estudiantes aprenderán como un conjunto de Ecuaciones Diferenciales Parciales (EDPs), que representan un comportamiento físico, se simula mediante Elementos, Volúmenes o Diferencias Finitas en un computador. El énfasis de la asignatura está en la generación de mallas geométricas, sobre todo, para el método de Elementos Finitos. De igual forma se entregan conocimientos básicos para los otros métodos numéricos mencionados

Objetivos

Entender cómo se modela un objeto en un Computador.

- Conocer las diferencias entre los distintos métodos de aproximación numérica.
- Conocer para que sirve una malla geométrica.
- Programar las técnicas más conocidas

Contenidos

1. Representación discreta de un objeto contínuo en un computador.
2. Modelando una interacción sobre una geometría simple.
3. Métodos de generación de mallas de Superficie y Volumen.
4. Definición de validez de un elemento.
5. Definición de calidad de un elemento.
6. Métodos de reparación y mejoramiento.

Metodología

Clases expositivas - Aprendizaje basado en la comprensión e implementación de algoritmos - Aprendizaje basado en estudio de autores - Estudio independiente y exposiciones de estudiantes acerca de temas - específicos - Al finalizar el semestre, los estudiantes deberán presentar un proyecto de fin de curso en el cual deberán leer, comprender, programar y exponer una publicación científica aprobada por el profesor.

Evaluación

Nota final = NC*0.6 + NP*0.4 En donde, NC es el promedio de dos certámenes y NP, es la nota del proyecto. Tanto NC como NP deben ser aprobados por separado.

Bibliografía

[1] Frey, P. & George, P. Mesh Generation: Applications to Finite Elements Hermes, Paris, 2000.
[2] Oden J., Tinsley R. & Junuthula N. An introduction to the mathematical theory of finite elements, New York : John Wiley, 1976.

Revistas

[1] Ito, Y., Shih, A., Soni, B.: Octree-based reasonable-quality hexahedral mesh generation using a new set of refinement templates. International Journal for Numerical Methods in Engineering 77(13), 1809–1833 (2009). [2] Bucki M., **Lobos C.,** Payan Y., *A Fast and Robust Patient Specific Finite Element Mesh Registration Technique: Application to 60 Clinical Cases.* Medical Image Analysis 14(3), 303–317 (2010).

Elaborado:	Claudio Lobos	**Observaciones:** **Actualización:** Diciembre 2012
Aprobado:	Depto. de Informática	
Fecha:	02/07/2010	

Asignatura	Métodos Cuantitativos en el Procesamiento Computacional de Imágenes (LSC)	**Sigla**	INF-569
Créditos SCT	4	**Conocimientos previos:** : ILI-280 Estadística Computacional ILI-286 Computación Científica 2 + autorización del Profesor	

Descripción

Asignatura centrada en los métodos matemáticos y computacionales aplicados al procesamiento de señales y, especialmente, imágenes. La asignatura introduce los elementos computacionales básicos del procesamiento de señales e imágenes, tales como los modelos vectoriales, matriciales y de campos tensoriales, las técnicas de filtrado y convolución, ciertos tipos de transformadas usuales en este contexto (Fourier, Hilbert, wavelets), y aplica estos formalismos a la solución de problemas de visualización, compresión, y procesamiento de señales e imágenes en general

Requisitos de Entrada:

Buen dominio de los elementos del Álgebra Lineal, Computación Científica 1 y 2.

Buen dominio de Matlab, C y sus aplicaciones.

Buena capacidad de comprensión de textos técnicos en inglés

Competencias:

1. Capacidad para identificar, plantear y resolver problemas.
2. Capacidad para aplicar los conocimientos en la práctica.
3. Capacidad de abstracción y modelado matemático y computacional.
4. Capacidad de diseño en ingeniería y sus aplicaciones.
5. Capacidad de investigación en ingeniería y sus aplicaciones.
6. Pensamiento sistémico y holístico.
7. Trabajar en equipos (multi-disciplinarios entre otros).
8. Capacidad de aprender por cuenta propia y actualizarse

Objetivos

1. Identificar problemas en el ámbito del procesamiento de señales e imágenes, susceptibles de ser representados adecuadamente mediante modelos computacionales con base en campos vectoriales, matriciales y tensoriales.
2. Construir modelos computacionales, como los mencionados, para abordar problemas y situaciones en el procesamiento de señales e imágenes donde ello es pertinente.
3. Seleccionar algoritmos, o eventualmente diseñar unos nuevos, que permitan resolver problemas en el ámbito del procesamiento de señales e imágenes mediante técnicas computacionales con base en el análisis lineal.
4. Contribuir a mejorar las competencias de los estudiantes en el ámbito del modelado y la resolución de problemas del procesamiento de señales e imágenes y, de este modo, contribuir también a incrementar la experiencia de los estudiantes en la redacción de documentos científicos

Contenidos

Contenidos temáticos	1	2	3
1.- Elementos del análisis tensorial.	**X**		**X**
2.- La transformada de Radon.		**X**	**X**
3.- Filtros y convolución		**X**	**X**
4.- Reconstrucción discreta de imágenes		**X**	**X**
5.- Técnicas algebraicas de reconstrucción de imágenes.		**X**	**X**

6.- Imágenes obtenidas mediante la técnica de resonancia magnética.		**X**	**X**		**X**
7.- Procesamiento de imágenes mediante técnicas con base en campos tensoriales.		**X**	**X**		**X**

Metodología

1. Clases expositivas con apoyo de medios visuales.
2. Exposiciones frecuentes de los participantes.
3. Aprendizaje con base en el desarrollo de tareas y de laboratorio.
4. Resolución de problemas y casos.
5. Discusiones.

Dedicación del Alumno:

Cátedra: 40 Hrs.
Estudio Personal 20 Hrs.
Redacción de Artículos 40 Hrs.
Desarrollo de Proyectos 20 hrs.
Total: 120 Hrs.

Evaluación

Evaluación	**Resultados del Aprendizaje**			
	1	**2**	**3**	**4**
Seminarios y exposiciones de los estudiantes	**X**	**X**	**X**	**X**
Desarrollo de artículos científicos	**X**	**X**	**X**	**X**

Calificación

Nota final = Promedio de todas las notas obtenidas por cada alumno en la asignatura.

Bibliografía

[1] T.G. Freeman, The mathematics of medical imaging. Springer, New York, 2010.
[2] G. Dougherty, Digital Image Processing for Medical Applications. Cambridge U. Press, New York, 2009
[3] S. G. Hoggar, Mathematics of Digital Images. Cambridge U. Press, Cambridge, 2006.
[4] A.K. Louis, F. Natterer, Mathematical problems of computerized tomography, Proc. IEEE 71 (1983), 379-389.
[5] F. Natterer, The mathematics of computerized tomography, SIAM, Philadelphia, 2001.
[6] M. Nixon, A. Aguado, Feature Extraction and Image Processing. Elsevier, Amsterdam, 2008 .
[7] L. O'Gorman, M.J. Sammon, M. Seul, Practical Algorithms for Image Analysis. Cambridge U. Press, New York, 2008.
[8] J. Weickert, H. Hagen, Visualization and Processing of Tensor Fields. Springer, New York, 2006

Elaborado:	Prof. Luis Salinas	**Observaciones:**
Aprobado:	Depto. de Informática	
Fecha:	20-08-2010	

Asignatura	Programación con Restricciones	**Sigla**	INF-571
Créditos SCT	8	**Conocimientos previos:** Optimización Combinatoria	

Descripción

El curso presenta los conceptos fundamentales de la programación con restricciones. Se estudia los problemas clásicos donde el modelamiento basado en la satisfacción de restricciones es aplicado. Se estudia los fundamentos de los métodos de resolución de problemas de satisfacción de restricciones. Los conceptos teóricos son complementados con el uso de herramientas computacionales para el modelamiento y la resolución de problemas de satisfacción de restricciones.

Objetivos

- Formular modelos basados en la satisfacción de restricciones y en la optimización con satisfacción de restricciones.
- Conocer las técnicas de resolución de problemas de satisfacción de restricciones.
- Conocer lenguajes para la programación con restricciones.

Contenidos

1. Problemas de satisfacción de restricciones (CSP): conceptos básicos, modelamiento de problemas basado en la satisfacción de restricciones
2. Técnicas de resolución de CSPs: enumeración exhaustiva, consistencia local, técnicas híbridas.
3. Problemas de optimización con satisfacción de restricciones (CSOP): conceptos básicos, modelamiento de problemas basado en la optimización con satisfacción de restricciones.
4. Técnicas de ramificación y acotamiento para la resolución de CSOPs: técnicas basadas en backtracking, técnicas basadas en la ramificación y acotamiento.
5. Lenguajes para la programación con restricciones: modelamiento y resolución de problemas industriales utilizando los diversos lenguajes.

Metodología

La asignatura contempla la realización de clases de exposición de los elementos teóricos, la realización de tareas individuales de aplicación de los conceptos vistos en clases y el estudio de casos y técnicas propuestas en artículos que presentan el desarrollo actual de la materia.

Evaluación

La asignatura se evalúa en base a certámenes (60%), análisis de artículos (20%) y tareas (20%).

Bibliografía

[1] Principles of Constraint Programming, Krzysztof Apt, Cambridge University Press, 2009.
[2] Handbook of Constraint Programming, Francesca Rossi, Peter van Beek & Toby Walsh, Elsevier, 2006.
[3] Constraint Processing, Rina Dechter, Morgan Kaufmann Publishers, 2003.

Elaborado:	Carlos Castro	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	1/8/2003	

Asignatura	Computación Evolutiva	**Sigla**	INF-572
Créditos SCT	8	**Conocimientos previos:** Inteligencia Artificial, programación en C (C++) para la implementación de las diversas estrategias.	

Descripción

En este curso se profundizan los conceptos de algoritmos de búsqueda local, algoritmos genéticos, algoritmos evolutivos, meméticos. Se hace énfasis en las aplicaciones y en los problemas de optimización combinatoria con restricciones.

Objetivos

Presentar al estudiante las técnicas basadas en la evolución como alternativas en la resolución de problemas de optimización y de satisfacción de restricciones.

- Presentar al estudiante los requerimientos formales de la aplicación de estas técnicas, tanto para su diseño e implementación, así como para su evaluación.

Mostrar al alumno el uso de la computación evolutiva en la resolución de problemas del mundo real.

Contenidos

1. Optimización Combinatoria: historia, problemas clásicos de optimización combinatoria con satisfacción de restricciones (CSOP), algoritmos para resolver problemas de satisfacción de restricciones (CSP), búsqueda del óptimo (técnicas basadas en Hill Climbing, Greedy Search, Tabu Search, Simulated Annealing).
2. Algoritmos Genéticos (AG): historia, y componentes de un AG (criterios para definir funciones de evaluación, criterios para definir operadores, noción de auto-adaptabilidad y tuning de parámetros). Algoritmos Híbridos, Co-evolución
3. Estrategias Evolutivas, el método de Schwefel
4. Programación Evolutiva, el método de Fogel & Fogel.
5. Swarm Intelligence
6. Investigaciones relacionadas en Computación Emergente
7. Conclusiones, investigación actual.

Metodología

Clases expositivas, seminarios, reuniones y presentaciones de avances del proyecto

Evaluación

El ramo implica la realización de un proyecto, con dos informes (20% y 30%), una presentación y una defensa (50%).

Bibliografía

[1] A Field Guide to Genetic Programming, R. Poli, W.B.Langdon, and N. Freitag, 2008
[2] Introduction to Evolutionary Computing, A. Eiben and J.E. Smith, Springer, 2010
[3] Bio-Inspired Artificial Intelligence, D. Floreano and C. Mattiussi, MIT Press, 2008
[4] Emergent Computing Methods in Engineering Design, D.E. Grierson and P.Hajela, 2010.
[5] Handbook of evolutionary Computation", T Bäck, Z. Michalewicz and D. Fogel, Oxford University Press, 1997.
[6] How to Solve It: Modern Heuristics, Z. Michalewicz and D. Fogel, Springer Ed., 1999
[7] Swarm Intelligence, R. Ebenhart, Y. Shui, J. Kennedy, Morgan & Kuffmann, 2001.
[8] Artículos en IEEE Transactions on Evolutionary Computation, Journal of Heuristics, Applied Soft Computing Complex Systems. Y Conferencias GECCO, IEEE CEC, PPSN.

Elaborado:	María Cristina Riff	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	1/8/2000	

Asignatura	Lógica Borrosa	**Sigla**	INF-572
Créditos SCT	8	**Conocimientos previos:** Lenguajes de programación, Cálculo Numérico, Estadística computacional, Álgebra	

Descripción

En este curso se dan los fundamentos formales de la Lógica Borrosa, a partir del contexto de la lógica matemática clásica, particularmente el cálculo de predicados. Los (sub)conjuntos borrosos emergen vinculados al lenguage natural con una intencionalidad semántica: por una parte representan el uso del lenguaje (en un contexto dado) y por otra, permiten el manejo formal de conocimientos vagos. Se discutirán distintos operadores para el cálculo con predicados borrosos, se presentarán teoremas de caracterización y mecanismos de generalización. El curso terminará con el análisis de inferencias en lógica borrosa.

Objetivos

Proporcionar un marco teórico formal para poder trabajar con lógica borrosa y poder desarrollar aplicaciones en p.ej. control automático borroso o sistemas expertos

Contenidos

1. Introducción; tipos de predicados
2. Funciones de E en [0,1]
3. t-normas, t-conormas y negaciones
4. Conjuntos borrosos
5. Consistencia lógica e inconsistencia en F (E)
6. Modificadores lingüísticos
7. Medidas de posibilidad
8. Funciones de Agregación
9. Principios de Extensión
10. Cortes alfa y números borrosos
11. Implicaciones borrosas
12. Regla Composicional de Inferencia
13. Cuantificadores borrosos

Metodología

Clases expositivas apoyadas con presentaciones en Powerpoint. Discusiones en grupo.

Evaluación

Tareas (25%); Trabajo final (75%)

Bibliografía

[1] Barro S., Bugarín A., Moraga C., Trillas E.: *Computación fuzzy*. En: "*Fronteras de la Computación*" (S. Barro, A. Bugarín, Eds.) Fundación Dintel – Díaz Santos, Madrid, ISBN 84-7978-517-9, 2002

[2] Klir G.J., St.Clair U.H., Yuan B.: *Fuzzy Set Theory*, Prentice Hall, 1997

[3] Mesiar R.: *Triangular Norms: an Overview*, In: *Computational Intelligence in Theory and Practice* (B. Reusch, K.-H. Temme Eds.) Physica Verlag, Heidelberg, 2001
[4] Moraga C., Trillas E., Guadarrama S.: Multiple-valued Logic and Artificial Intelligence. Fundamentals of Fuzzy Control Revisited. *Journal of Artificial Intelligence*, **20** (3-4) 169-197, 2003 (*)
[5] Moraga C.: Introduction to Fuzzy Logic. *Facta Universitatis*, Serie E.E., (University of Niš, Serbia), **18** (2), 2005 (*)
[6] Nguyen H.T., Walker E.A.: *A first Course in Fuzzy Logic*. CRC Press, Boca Raton FLA, 1997
[7] Tanaka K.: *An Introduction to Fuzzy Logic for Practical Applications*, Springer, Berlin, 1997
[8] Trillas E., Cubillo S.: Lecciones de Lógica Borrosa. Facultad de Informática, Universidad Politécnica de Madrid, 1998

Elaborado:	C. Moraga	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	16-03-2010	

Asignatura	Máquinas del Aprendizaje Computacional	**Sigla**	INF- 578
Créditos SCT	8	**Conocimientos previos:** ILI-280; MII-475	

Descripción

En este curso se presenta un tratamiento sistemático de los fundamentos del aprendizaje estadístico inductivo, deductivo y transductivo, abordando temas como el riesgo de generalización, minimización del riesgo estructural y los dilemas sesgo-varianza y estabilidad-flexibilidad. También se estudian diversas metodologías y técnicas para el diseño y construcción de algoritmos de aprendizaje supervisado, tales como las Redes Neuronales Artificiales (ANN), las Máquinas de Vectores de Soporte (SVM), y Algoritmos de Aprendizaje No- Supervisado tales como las Redes de Kohonen o Mapas Auto-Organizativos (SOM) y el Algoritmo Vecino más Cercano (NN) con sus variantes más relevantes, incluyendo también el razonamiento aproximado basado en diversos métodos de agregación de máquinas (*ensembles*). Todo lo anterior inmerso en el contexto de problemas de reconocimiento de formas, tales como clasificación, regresión, agrupamiento y pronóstico. Además se discuten algunos temas de reciente interés tales como el aprendizaje incremental, el aprendizaje distribuido, y la búsqueda de relaciones entre el aprendizaje de máquinas y el aprendizaje humano.

Objetivos:

Al finalizar el curso el alumno estará capacitado para:

- Conocer los fundamentos estadísticos y computacionales de las Máquinas de Aprendizaje.
- Diseñar y aplicar Máquinas de Aprendizaje a problemas de reconocimiento de formas: Clasificación, Asociación, Pronóstico, entre otras.
- Conocer y aplicar diversos algoritmos de Máquinas de Aprendizaje.
- Conocer los principales avances de las Artificial Neural Networks (ANN), para los problemas de clasificación y reconocimiento de patrones.

Contenidos:

1. Conceptos estadísticos en el aprendizaje: problemas de Inducción e Inferencia Estadística. Modelos de inferencia Paramétrica y no Paramétrica. Generalización de la teoría de Glivenko – Cantelli – Kolmogorov.
2. Teoría de aprendizaje y generalización: el Problema de minimización del riesgo en datos empíricos y el dilema Sesgo-Varianza. El Problema de reconocimiento de formas. El problema de estimación en regresión generalizada. El Problema de estimación de la densidad, Teorema de Glivenko y Cantelli, tipos de convergencia, Ley de Kolmogorov – Smirnov y Ley de los Algoritmos Iterados.
3. Estimación de la densidad de probabilidad y el problema de aprendizaje: convergencia y condiciones de convergencia para medidas de probabilidades desconocidas.
4. Riesgo empírico y principios de minimización: Teorema de Key de la Teoría de Aprendizaje. Cotas y funciones de pérdidas.
5. El Perceptrón y sus generalizaciones: El Perceptrón de Rosenblatt. El método estocástico de aproximación.
6. Elementos de teoría de optimización teorema de Fermat: la regla de multiplicadores de Lagrange y la teoría de Kühn – Tucker.
7. Redes Neuronales en regresión generalizada.
8. La Máquina de Vector Soporte (SVM, Support Vector Machine): aproximación por hiperplanos, las propiedades estadísticas de los hiperplanos óptimos, generalización a espacios

de alta dimensión. Selección de SVM. SVM en reconocimiento de formas. SVM en problemas de clasificación múltiple. Aplicaciones.

Metodología de Trabajo:
El Curso se desarrollará con Clases expositivas de 90 minutos de exposición. El alumno deberá rendir interrogaciones y tareas individuales para complementar su aprendizaje. Durante el desarrollo del curso el alumno deberá seleccionar lecturas relevantes de artículos y elaborar un proyecto de investigación en algún tópico de su elección en forma de seminario y discusión del estado del arte. El examen final del curso estará basado en una presentación oral y un reporte escrito de su tema de investigación.

Evaluación:

Tareas e interrogaciones	30%
Proyecto	40%
Examen	30%

Bibliografía:
[1] Vladimir N. Vapnik. Statistical Learning Theory. Wiley, New York, 1998.
[2] B. Schölkopf, A.J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning). MIT Press, 2001.
[3] E. Alpaydin, Introduction to Machine Learning, Second edition. The MIT Press, 2009.
[4] T. M. Mitchell, Machine Learning, 1.ª ed. McGraw-Hill Science/Engineering/Math, 1997.
[5] C. M. Bishop, Pattern Recognition and Machine Learning, 1st ed. 2006. Corr. 2nd printing 2011. Springer, 2007.
[6] B. D. Ripley, "Pattern recognition and Neural Network", .Ed. Cambridge University Press, 1996.
[7] T.L. Fine, "Feedforward Neural Network Methodology", Ed. Springer Verlag, 1999.
[8] Richard O. Duda, Peter E. Hart, David G. Stork, Pattern Classification, Wiley, 2001.
[9] **Revistas**: Journal of Machine Learning, IEEEE Trans. on Neuronal Networks, Neuronal computing, Pattern Recognition, Journal of Intelligent Data Analysis, entre otras.

Material complementario en línea:
[1] T. Poggio, L. Rossaco, C. Fronger, y G. Canas, «Statistical Learning Theory and Applications, Spring 2012». [Online]. Available: http://www.mit.edu/~9.520/. [Accessed: 27-dic-2012].
[2] A. Ng, «Machine Learning | Coursera». [Online]. Available: https://www.coursera.org/course/ml. [Accessed: 27-dic-2012].
[3] T. Jaakkola, «Machine Learning MIT OpenCourseware», MIT OpenCourseWare. [Online]. Available: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/. [Accessed: 27-dic-2012].
[4] P. Domingos, «Machine Learning | Coursera». [Online]. Available: https://www.coursera.org/course/machlearning. [Accessed: 27-dic-2012].

Elaborado:	Héctor Allende	**Observaciones:**
Aprobado:	Depto. de Informática	**Actualización:** Diciembre 2012
Fecha:	01-08-2006	

Asignatura	Seminario de Especialidad I: Experimentación en Ingeniería de Software	**Sigla**	INF- 581
Créditos SCT		**Conocimientos previos:** Ingeniería de Software	

Descripción
Experimentación en Ingeniería de Software concierne el uso del diseño y análisis experimental para validar ideas y creencias, en un campo ampliamente dominado por suposiciones y especulaciones, orientado en forma práctica para los ingenieros de software. ¿Son válidas nuestras suposiciones? ¿Qué afirmaciones de la comunidad de desarrollo de software son válidas? ¿Bajo qué circunstancias son válidas? Responder estas preguntas es crítico para otorgar mayor certeza a las ideas en que se fundamenta la Ingeniería de Software. Durante la construcción de software no se utilizan, habitualmente, técnicas formales de experimentación. Este hecho contrasta con las prácticas comunes de otras ingenierías y campos científicos, en las cuales es obligatorio realizar una rigurosa experimentación que apoye las investigaciones realizadas. En este curso se discute el uso de Análisis y Diseño de Experimentos en Ingeniería de Software, estableciendo las bases teóricas para la efectiva realización de experimentos.

Objetivos:
Al finalizar el curso el alumno:
1. Comprenderá la importancia de validar empíricamente las técnicas usadas en el desarrollo de software
2. Sabrá aplicar diversas técnicas de experimentación en Ingeniería de Software
3. Conocerá algunas experiencias reales de experimentación en Ingeniería de Software
4. Conocerá líneas de investigación actuales y futuras en Ingeniería de Software Experimental

Contenidos
1. Introducción a la experimentación en Ingeniería de Software
2. Principales tipos de diseños y análisis de experimentos
3. Estudios experimentales reales realizados en Ingeniería de Software (diseño, ejecución, análisis crítico)
4. Estado del arte en Ingeniería de Software Experimental (líneas de investigación actuales y futuras, desafíos de la disciplina)

Metodología:
- Clases tipo seminario de 90 minutos, combinando exposición con discusión grupal
- Lectura sistemática guiada de artículos fundamentales y capítulos de libros, elaboración de comentarios semanales
- Selección de un tópico de profundización, selección de bibliografía relevante, elaboración de informes periódicos quincenales e informe final
- Diseño, ejecución y análisis crítico de estudios experimentales en Ingeniería de Software

Evaluación:
- Informes periódicos/comentarios semanales/quincenales: 30%
- Informe final: 30%
- Presentación oral: 30%
- Participación: 10%

Bibliografía:
[1] Basics of Software Engineering Experimentation. Natalia Juristo, Ana María Moreno. Kluwer, 2001

[2] Experimentation in Software Engineering: An Introduction. Claes Wohlin, Per Runeson, Martin Höst. Springer, 1999
[3] Empirical Software Engineering Issues. Vic Basili et al (editors). Lecture Notes in Computer Science 4336. Springer-Verlag, 2007

Conferencias

[1] Empirical Software Engineering and Measurement (ESEM)

Revistas

[1] Journal of Empirical Software Engineering (ESE)
[2] IEEE Software

Elaborado:	M. Visconti	**Observaciones:**
Aprobado:	Depto. de Informática	
Fecha:	23.04.08	

Asignatura	Seminario de Especialidad II: Cloud Computing y Big Data	**Sigla**	INF- 582
Créditos SCT	4	**Conocimientos previos:** Programación en C, Sistemas Operativos, Redes.	

Descripción

La asignatura introduce los conceptos de la computación en la "nube" como una plataforma alternativa para High Performance Computing (HPC), desde los elementos que componen un data center hasta los requerimientos de redes de interconexión, capacidades de procesamiento (computing) y capacidad de almacenamiento (storage). Se presentan las ventajas y desventajas del modelo en la "nube" para resolver aplicaciones técnicas y científicas que utilizan grandes volúmenes de datos (Big Data), que no son fáciles de mover. Se pone un especial énfasis en las soluciones costo/efectivas analizando, discutiendo y estableciendo comparaciones entre distintas soluciones de "nubes": pública y privadas, comerciales y académicas, así como su análisis tipo Total Cost of Ownership (TCO).

La asignatura contempla también un acercamiento práctico al tema a través de la construcción de aplicaciones en la nube como servicios, MapReduce (Hadoop), PaaS (Azure), mover grandes volúmenes de datos (RDBMS tradicional) o mover los algoritmos hacia los datos (NoSQL).

Este curso está dedicado a la programación paralela en la "nube". El objetivo del curso es ser capaz de modelar, diseñar, e implementar una solución paralela, cuyos resultados obtenidos deben ser difundidos a través de la escritura de un artículo técnico que debe ser presentado en el curso.

Objetivos:

- Presentar las arquitecturas modernas para la programación paralela y distribuida.
- Conocer problemas paralelizables y métodos para resolverlos.
- Paralelizar aplicaciones con Message Passing Interface (MPI).
- Saber programar una aplicación de Big Data para resolver problemas intensivos en cálculo.
- Presentar resultados en un artículo técnico.
- Conocer las ventajas y limitaciones de cloud computing.
- Relacionarse con problemas de gran complejidad y computación intensiva.

Contenidos

1) Introducción: Perspectiva histórica, nomenclatura, conceptos básicos, fuentes de grandes volúmenes de datos, ejemplos.
2) Conceptos Teóricos: Ley de Amdahl, Ley de Gustafson, speedup, Clase de Problemas de Nick (NC), Memoria distribuida y/o compartida, Algoritmos y complejidad con Big Data.
3) Modelos Teóricos: Modelos secuenciales y paralelos, PRAM, BSP en la nube, MPI, MapReduce. Ejemplos comerciales: Hadoop como arquitectura de cluster.
4) Diseño de algoritmo paralelo: Mappers and Reducers, Sistema de Archivo distribuido (HDFS), agregación local, pairs and stripes, etc.
5) Algoritmos y aplicaciones con Big Data: Algoritmos sobre grafos (grandes), datos de sensores, geo-referenciación, datos de redes sociales, etc.a. Máquinas de Soporte Vectorial transductivas

Metodología:

La metodología usada para el curso serán clases expositivas, investigación bibliográfica, trabajo práctico de programación en aplicación a elegir, escribir artículo, presentación técnica de los resultados.

Evaluación:
La nota final del curso de calculará de la siguiente forma:

- Un certamen a mediados del semestre: 30%
- Redacción de un artículo: 40%
- Presentación del artículo: 30%.

Bibliografía:

[1] Catlett Ch., Gentzsch W., Grandinetti L., Joubert G. and J. Vazquez-Poletti (Eds.), Cloud Computing and Big Data. IOS Press BV, Netherlands, 2013.
[2] De Bosschere K., D'Hollander E., Joubert G., Padua D. and F. Peters (Eds.), Applications, Tools, and Techniques on the Road to Exascale. IOS Press BV, Netherlands, 2012.
[3] Aguilar, L. Big Data: Análisis de Grandes Volúmenes de Datos en Organizaciones. Alfaomega. 2014.
[4] Grama A., Gupta A., Karypis G., Kumar V., Introduction to Parallel Computing, Second Edition, Addison Wesley, 2003.
[5] White, T., Hadoop, the definitive guide, O'Reilly, 3rd Ed., 2012.

Elaborado:	Mauricio Solar	**Observaciones:**
Aprobado:	Comité IP del DI	
Fecha:	23-02-2015	

Asignatura	Seminario de Especialidad II: Computación Cuántica	**Sigla**	INF- 582
Créditos SCT	4	**Conocimientos previos:** Necesarios:Algebra Lineal, Probabilidades y Estadística Deseables: Diseño de Circuitos Lógicos, Informática Teórica	

Descripción
El curso está destinado a estudiantes de Informática, sin conocimientos profundos de Física Teórica. El mundo cuántico sera modelado como un espacio de Hilbert sobre los complejos. Operaciones elementales sobre vectores cuánticos serán realizadas mediante transformaciones unitarias y visualizadas mediante la Esfera de Bloch. La meta del curso es discutir el diseño de circuitos computacionales y algoritmos en este mundo "extraño“ y aun poco conocido. No es meta del curso discutir algunos de los fascinantes y candentes problemas de la mecánica cuántica.

Objetivos:
Al término del curso el alumno deberá ser capaz de:
- Conocer los conceptos fundamentales de la Computación Cuántica.
- Conocer y aplicar un modelo basado en espacios de Hilbert para el estudio de Computación Cuántica
- Diseñar circuitos computacionales reversibles / cuánticos
- Especificar, analizar y diseñar algoritmos para un entorno de Computacíón Cuántica.

Contenidos
- ¿Por qué ocuparse de computación cuántica?
- Computación reversible
- El bit cuántico o „qubit“
- La notación de Dirac
- Estados básicos, superposición de estados, acoplamiento de estados
- Medición de observables
- La Esfera de Bloch
- Matrices de Pauli, transformación de Hadamard
- Modelo computacional cuántico
- Compuertas cuánticas básicas: NO, NO-controlado, Toffoli, Deutsch
- Compuertas universales, Arquitecturas del grupo Barenco y colegas.
- ¿Como realizar un mundo discreto confiable en un universo continuo?
- Ejemplos computacionales
- Complejidad clásica y complejidad cuántica
- Algoritmos cuánticos, aspectos de diseño, análisis de los más famosos
- (Si el tiempo lo permite) Corrección automática de errores
- (Si el tiempo lo permite) Lógica cuántica

Metodología:
El Curso se desarrollará con Clases expositivas de 90 minutos, cnn presentaciones powerpoint. El alumno deberá rendir tareas individuales para complementar su aprendizaje. Se contempla además un proyecto final, desarrollado individualmente. El proyecto final del curso será evaluado en base a un reportaje escrito sobre el tema de investigación acordado con el Profesor.

Evaluación:
- Tareas 30%
- Proyecto Final 70%

Bibliografía:
El curso comprende una selección de temas con una orientación similar a la del libro "Quantum Computation and Quantum Information," Cambridge University Press, 2000, de M.A. Nielsen e I.L. Chuang.
Los participantes recibirán Apuntes de Clase especialmente elaborados para el curso, como también apuntes de clase utilizados en CalTech y en la Universidad de Indiana además de una colección de artículos que amplían los aspectos presentados en el curso (Habrá un CD con todo este material, que puede ser copiado por los participantes para su uso personal.)
Adicionalmente, los participantes tendrán acceso a trabajos presentados en los recientes Workshops Internacionales realizados en Europa, sobre Computación Reversible / Computación Cuántica.

Elaborado:	Claudio Moraga Julio 2005	**Observaciones:** este curso está enfocado a estudiantes de Doctorado en Ingeniería, especialmente (pero no exclusivamente) para los estudiantes del Doctorado en Informática, y se basa en un alto compromiso de los estudiantes por aportar con su trabajo al curso y participar activamente en las clases.
Aprobado:	Comité IP del DI	
Fecha: **Actualizado:**	02-08-2005 Diciembre 2012	

Asignatura	Sistemas complejos discretos	**Sigla**	INF- 584
Créditos SCT	4	**Conocimientos previos:** Teoría de grafos, lenguajes formales, autómatas	

Descripción
Este curso presenta una panorámica de los sistemas complejos discretos, desde los sistemas más homogéneos (autómatas celulares), donde es posible obtener más resultados teóricos exactos, hasta las redes heterogéneas y masivas, con las que debemos interactuar sin jamás llegar a conocerlas por completo; en el camino se estudian sistemas dinámicos definidos sobre grafos pequeños (en particular, redes booleanas), y se ilustran repetidamente los fenómenos emergentes y la dificultad de análisis que caracterizan a los sistemas complejos, ya desde sus instancias supuestamente más simples. Se pone énfasis en las aplicaciones al campo bioinformática (pasando revista también a otras áreas de aplicación), aunque las herramientas algorítmicas y la perspectiva amplia sobre el tipo de fenómenos son aplicables en una gran cantidad de áreas.

Objetivos:
Al aprobar la asignatura el alumno tendrá una perspectiva teórica sólida y amplia de los sistemas discretos masivos, presentes hoy en día en casi toda área del quehacer; estará familiarizado con los fenómenos dinámicos y problemas analíticos que aparecen como consecuencia de la complejidad de las interacciones y naturaleza de los agentes que interactúan, y conocerá las formas actuales de abordar los principales problemas que se presentan. Habrá trabajado con algoritmos y habrá debido enfrentar desafíos teóricos y prácticos en sistemas con diversos niveles de complejidad estructural y dinámica. Podrá conectar estos conocimientos con diversos sistemas de importancia social, científica y tecnológica, y en prácticamente todos los temas habrá conocido investigación de punta (y muchas preguntas aún por contestar).

Contenidos

1. Autómatas celulares. Ejemplos. Estudio teórico. Subshifts y lenguajes formales asociados. Universalidad, reversibilidad, problemas duros y problemas indecidibles. Clasificaciones fenomenológicas y analíticas. Complejidad comunicacional, flujo de información, física emergente del procesamiento de información. Relación con modelos continuos y con modelos basados en agentes; autómatas conservativos y modelos de tráfico. Usos: modelos físicos, biológicos, sociales, epidemiológicos; procesamiento de imágenes, criptografía, protocolos de sincronización. Autómatas celulares estocásticos. Autómatas cuánticos (QCA). Fenómenos emergentes; estructuras replicativas. Pilas de arena, criticalidad auto-organizada.

2. Conceptos generales de sistemas complejos discretos. Nociones de sistemas dinámicos, atractores, sensibilidad, no-linearidad, dinámicas caóticas. Fenómenos emergentes, nociones de complejidad.

3. Redes de autómatas: dinámicas sobre grafos de tamaño moderado, con énfasis en redes booleanas. Redes neuronales discretas, circuitos lógicos; modos de iteración. Convergencia, funcionales de Lyapunov. Robustez dinámica frente a ruido y cambios en modos de iteración. Dependencia de dinámica respecto a la topología: modelos de evolución y cooperación en grafos; juegos iterativos. Problemas algorítmicos asociados a redes booleanas; estudio especial de la aplicación a redes de regulación genética. El problema inverso de determinación de redes a partir de información parcial. Redes booleanas aleatorias (RBN) como modelos de dinámicas biológicas.

4. Modelos de grafos aleatorios para sistemas de gran tamaño. Modelos clásicos (Erdös-Renyi). Estadísticas sobre redes: conexidad, patrones locales, interacción de características estructurales locales y globales.

5. Redes complejas. Topologías principales: scale-free, small worlds. Propiedades: autosimilaridad, clusterización, diámetro pequeño. Algoritmos de construcción de redes; mecanismos generativos espontáneos. Inferencia de propiedades estructurales en grandes redes; ley de potencia de grados. Navegabilidad y búsquedas descentralizadas en redes complejas. Representación gráfica. Detección de clusters y comunidades. Minería de datos de redes. Análisis de roles, análisis de enlaces. Robustez estructural antes fallas o ataques; mecanismos de evolución estructural. Sistemas dinámicos en redes complejas: autoorganización, fenómenos de contagio, estrategias de control dinámico y estructural. Estabilidad dinámica, acoplamientos de oscilaciones, correlaciones y flujo de información en redes. La aplicación de las ideas será con énfasis en redes complejas biológicas (nerviosas, metabólicas, regulatorias, interacciones de proteínas) pero con atención a las particularidades de redes complejas sociales, semánticas y tecnológicas (como la Web).

Metodología:
El Curso se desarrollará con Clases expositivas de 90 minutos de exposición. El alumno deberá rendir interrogaciones y tareas individuales para complementar su aprendizaje. Durante el desarrollo del curso el alumno deberá seleccionar lecturas relevantes de artículos y elaborar un proyecto de investigación en algún tópico de su elección en forma de seminario y discusión del estado del arte. El examen final del curso estará basado en una presentación oral y un reporte escrito de su tema de investigación.

Evaluación:

Tareas e interrogaciones	30%
Proyecto	40%
Examen	30%

Bibliografía:
[1] Boccara, N. Modeling Complex Systems. Springer Verlag, 2004.

[2] Wolfram, S., Ed. Theory and applications of cellular automata. World Scientific Publication, 1986.

[3] Brandes, U., Erlebach, T., Eds. Network Analysis: Methodological Foundations (LNCS 3418), Springer-Verlag, 2005.

[4] Bornholdt, S., Ed. Handbook of Graphs and Networks: From the Genome to the Internet. Wiley-VCH, 2003.

[5] Junker, B., Schreiber, F., Eds. Analysis of Biological Networks. John Wiley & Sons, 2008.

Elaborado:	A. Moreira Julio 2008	**Observaciones:**
Aprobado:	Depto. de Informática	
Fecha:	Julio 2008	

Asignatura	Base de Datos Documentales	**Sigla**	INF- 585
Créditos SCT	4	**Conocimientos previos:**	

Objetivos: Al finalizar el curso el alumno estará capacitado para:
1. Conocer los conceptos fundamentales de recuperación de información
2. Conocer los modelos de recuperación de información más usados en bases de datos documentales
3. Conocer las estructuras de datos usadas con mayor frecuencia en la implementación de sistemas de bases de datos documentales
4. Conocer las nociones fundamentales para búsqueda e indexamiento de documentos en la web.
5. Conocer los principales avances y temas de investigación en recuperación de información.

Contenidos
1. Propiedades y operaciones en texto: crecimiento, dispersión, distribuciones, metadatos v/s texto, formatos, nociones de teoría de información, preprocesamiento de texto, stemming, tesauros.
2. Modelos de recuperación de información: modelo booleano, vectorial, probabilístico, fuzzy, booleano extendido, vectorial generalizado, LSI, ANN, Bayesiano.
3. Evaluación de modelos de recuperación de información - métricas y metodologías: precisión, cubrimiento, F-medida.
4. Estructuras de datos para texto: archivos invertidos, árboles de sufijos, arreglos de sufijos, archivos de firma.
5. Recuperación de información en la web: estructura de buscadores (crawlers, índices, ranking en la Web). Uso de hyperlinks, texto, logs: algoritmos HITS, PageRank, Bayesiano, clustering de consultas, web mining.
Temas avanzados: motores de búsqueda 3G, recuperación de información en música, recuperación de información en video, web semántica.

Metodología:
El Curso se desarrollará con clases expositivas de 90 minutos de exposición. El alumno deberá rendir una evaluación escrita a mitad del semestre, realizar una presentación de un artículo de su interés sobre algún tema avanzado de investigación y desarrollar un mini-proyecto con aporte creativo. El examen final del curso será basado en una presentación oral y en un informe escrito de su mini-proyecto.

Evaluación:
Certamen: 30%, Presentación: 30%, Examen: 40%

Bibliografía:
[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley, 1999.

BIBLIOGRAFÍA COMPLEMENTARIA
[1] Witten, I., Moffat, A. y Bell, T. Managing Gigabytes, Morgan Kauffman, 1999 (segunda edición).
[2] Richard O. Duda, Peter E. Hart, David G. Stork, Pattern Classification, Wiley, 2001
[3] Agosti, M. y Smeaton, A. (editores) Information Retrieval and Hypertext, Kluwer, 1996.
[4] Revistas: Journal of the American Society for Information Science and Technology, Journal of Information Retrieval, Pattern Recognition, Journal of Intelligent Data Analysis, etc.

Elaborado:	Marcelo Mendoza Rocha	**Observaciones:**
Aprobado:	Depto. de Informática	
Fecha:	23.04.08	

Asignatura	Seminario de Tesis I	**Sigla**	INF- 591
Créditos SCT	4	**Prerrequisitos:** Examen de calificación aprobado	

Descripción
Seminario destinado a estudiar en la literatura especializada, los trabajos relacionados con el tema de tesis del alumno.

Objetivos
Profundizar en el tema específico de la tesis, explorando las fronteras del conocimiento y proponiendo métodos alternativos.

Contenidos
Variable

Metodología
Presentación y discusión de trabajos en forma de seminario y presentación en congresos de especialistas

Evaluación
Obtiene una nota al aprobar el tema de tesis en su examen de calificación.

Bibliografía
Variable

Elaborado:	Prof. Guía de Tesis	**Observaciones:** Revisado Enero 2012.
Aprobado:	Depto. de Informática	
Fecha:	23.04.08	

Asignatura	Seminario de Tesis II	**Sigla**	INF- 592
Créditos SCT	4	**Prerrequisitos:** Examen de calificación aprobado	

Descripción
Seminario destinado a estudiar en la literatura especializada, los trabajos relacionados con el tema de tesis del alumno.

Objetivos
Profundizar en el tema específico de la tesis, explorando las fronteras del conocimiento y proponiendo métodos alternativos.

Contenidos
Variable

Metodología
Presentación y discusión de trabajos en forma de seminario y presentación en congresos de especialistas

Evaluación
Obtiene una nota al aprobar el tema de tesis en su examen de calificación.

Bibliografía
Variable

Elaborado:	Prof. Guía de Tesis	**Observaciones:** Revisado Enero 2012.
Aprobado:	Depto. de Informática	
Fecha:		